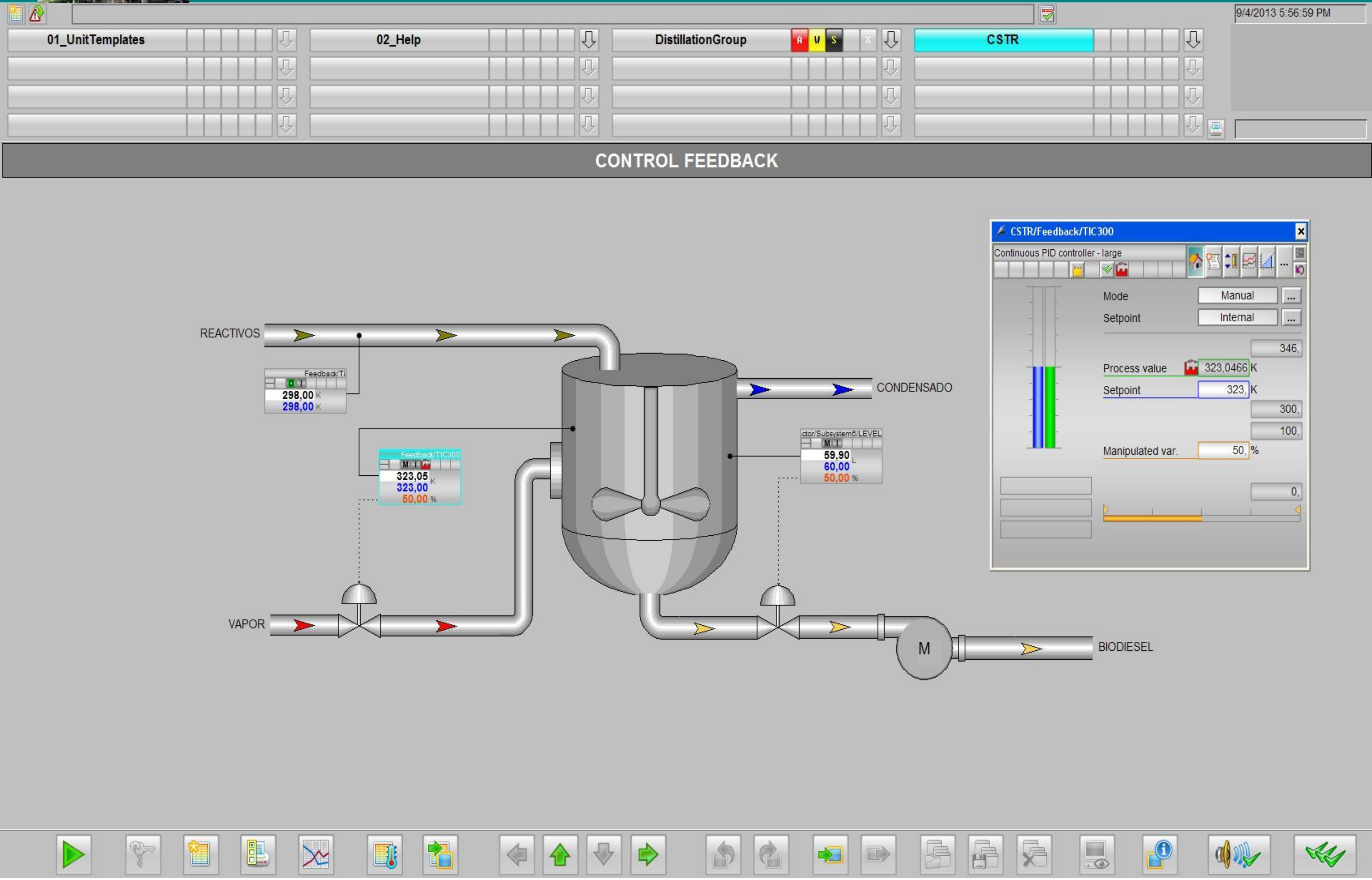
Diseño controladores en el Dominio Discreto



Objetivos Generales

- Diseñar controladores basados en herramientas del Dom. Discreto
  - Procedimientos alternativos:
    - Diseñar en Dom. Frec. (TP2) y luego discretizar → equivalentes discretos
    - Utilizar técnicas de transformación (Transformada W) bajo hipótesis
    - Desarrollar directamente el controlador en Dom. discreto (mét. de Ragazzini)

- Familiarizarse con herramientas computacionales (funciones de Matlab)
  - Cálculo de equivalentes discretos de funciones continuas
    - `"c2d"`, `"recadel"`, `"recatras"`
  - Cómputo de la transformada W (y su inversa)
    - `"transfw"`, `"antitransfw"`

Objetivos específicos: 4 secciones

- 1. Cálculo y simulación de equivalentes discretos (sección 2.1)
  - Definir un sist. discreto con = características (en D.T. y D. F.) que su equiv. continuo
  - Métodos de integración numérica → reglas rectangulares y trapezoidal
  - Método de Mapeo de polos y ceros → relación $z = e^{sT}$ entre planos s y z
  - Método del bloqueador equivalente $$H(z) = (1 - z^{-1})Z\left[\frac{H(s)}{s}\right]$$
    - Supone la presencia de un bloqueador de orden 0
- 2. Diseño de un controlador digital usando técnicas de transformación (ejercicio integrado en la sección 2.2.2)
  - Método de la Transformada W (procedimiento e hipótesis de aplicación)
- 3. Diseño de un controlador por el método de Ragazzini (sección 2.2.1)
  - Método directo a partir de las especificaciones y restricciones de diseño
- 4. Diseño de un controlador utilizando un Compensador de Smith (sección 2.2.2)
  - Compensar tiempo muerto del proceso (como en TP1)

Material disponible TP3

- Campus virtual
  - Enunciado TP
  - Presentación (.pdf)
  - Archivos Matlab
    - Simulación
      - Modelo no lineal (Simulink) → “CSTR_TP3_2021.mdl”
      - parametrización modelo → “data.mat”
    - Scripts
      - Cálculo equiv. Discretos
        - `“c2d”, “recadel”, “recatras”`
      - Cómputo de la Transformada W y su anti-transformada
        - `“transfw”, “antitransfw”`

2.1. Cálculo y simulación de equivalentes discretos

- Cálculo del equiv. discreto del controlador *J(s)* (PI+PD) diseñado en TP2
  - Utilizando los diferentes métodos (5)
  - Utilizando diferentes períodos de muestreo (T=2 y T=20)
  - Estudio de la estabilidad
    - ¿Qué ocurre con los polos de los equiv. discretos calculados?
  - Análisis y comparación de las características en D. Frecuencial
    - Bode magnitud
  - Análisis y comparación de la rta. Temporal
    - Servo-comportamiento
    - "CSTR_TP3_2021.mdl" y "data.mat"
  - Discrepancias para T=20 → ¿Por qué T=20 no es un periodo de muestreo adecuado?
    - ¿Relación con la constante de tiempo más rápida del sistema?

- Partiendo de *J(s)* en formato "*zpk*"
  - Regla trapezoidal → `Jz = c2d(J, T, 'tustin')`
  - Regla rectangular por adelanto → `Jz = recadel(J, T)`
  - Regla rectangular por atraso → `Jz = recatras(J, T)`
  - Mapeo de polos y ceros → `Jz = c2d(J, T, 'matched')`
  - Bloqueador Equivalente → `Jz = c2d(J, T, 'zoh')`

2.1. Cálculo y simulación de equivalentes discretos

- Cálculo del equiv. discreto del controlador *J(s)* (PI+PD) diseñado en TP2
  - Utilizando los diferentes métodos (5)
  - Utilizando diferentes períodos de muestreo (T=2 y T=20)
  - Estudio de la estabilidad
    - ¿Qué ocurre con los polos de los equiv. discretos calculados?
  - Análisis y comparación de las características en D. Frecuencial
    - Bode magnitud
  - Análisis y comparación de la rta. Temporal
    - Servo-comportamiento
    - “CSTR_TP3_2020.mdl” y “data.mat”
  - ¿Por qué T=20s. no es un periodo de muestreo aceptable?
    - Relación con la cte de tiempo más rápida?

- Partiendo de *J(s)* en formato “*zpk*”
  - Regla trapezoidal → `Jz = c2d(J, T, ‘tustin’)`
  - Regla rectangular por adelanto → `Jz = recadel(J, T)`
  - Regla rectangular por atraso → `Jz = recatras(J, T)`
  - Mapeo de polos y ceros → `Jz = c2d(J, T, ‘matched’)`
  - Bloqueador Equivalente → `Jz = c2d(J, T, ‘zoh’)`

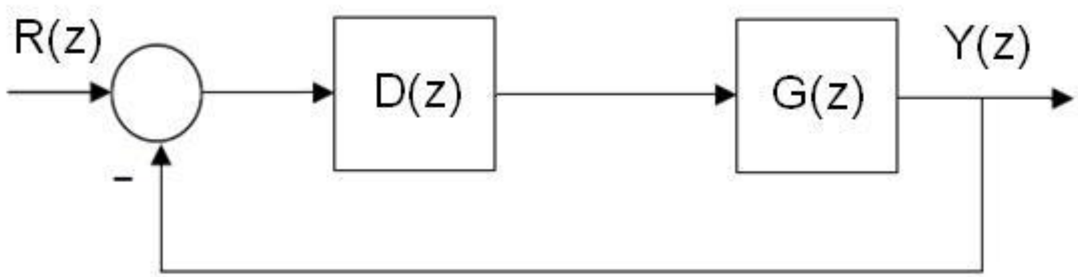
2.2.1. Método de diseño directo: Ragazzini

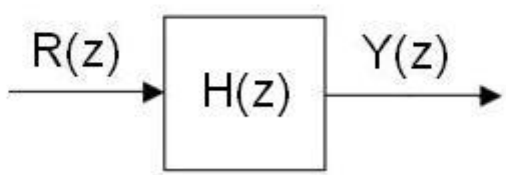
- Se dispone modelo del proceso G(z)
- Se asume estructura con controlador D(z) y realimentación unitaria
- Se desea una determinada dinámica H(z) entre R(z) e Y(z)

$$H(z) = \frac{DG}{1 + DG}$$

- Objetivo: diseñar D(z) para lograr la H(z) deseada

$$D(z) = \frac{1}{G(z)} \frac{H(z)}{1 - H(z)}$$

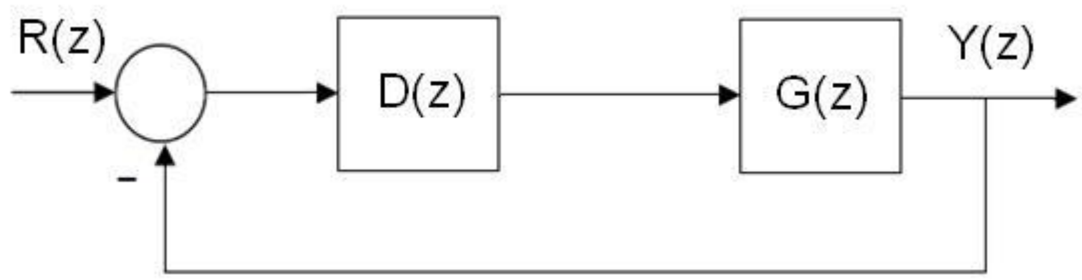


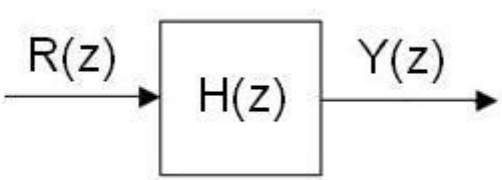


2.2.1. Método de diseño directo: Ragazzini

$$D(z) = \frac{1}{G(z)} \frac{H(z)}{1 - H(z)}$$

- D(z): busca cancelar dinámica del proceso y agregar dinámica deseada
- Existen restricciones sobre H(z) para que el problema sea factible





2.2.1. Ragazzini: Limitaciones sobre H(z)

- **1. Precisión dinámica**
  - La dinámica requerida está asociada con la ubicación de los polos de H(z)
  - Asumimos dinámica de 2do orden
  - Denominador de H(z) es: $z^2 - (2r\cos\theta)z + r^2$ con: $r = e^{-pw_nT}$

    $\theta = w_n T\sqrt{1-p^2}$

- **2. Causalidad de D(z)**
  - Para que D(z) sea causal → el orden(denominador) ≥ orden(numerador) (i.e. D(z) no debe tener polos en infinito)
  - Dado que $D(z) = \dfrac{1}{G(z)}\dfrac{H(z)}{1-H(z)}$ entonces:
    - Si G(z) tiene un nro de ceros en inf, H(z) debe tener el mismo nro de ceros en inf (caso contrario D(z) tendría polos en inf)
  - $F(z) = \dfrac{az^m + bz^{m-1} + \cdots}{cz^n + dz^{n-1} + \cdots}$ *F(z)* tiene *(n-m)* ceros en Inf. si *n>m* $\lim_{z\to\infty} F(z) = 0$
  - Ya tenemos la estructura de H(z)

2.2.1. Ragazzini: Limitaciones sobre H(z)

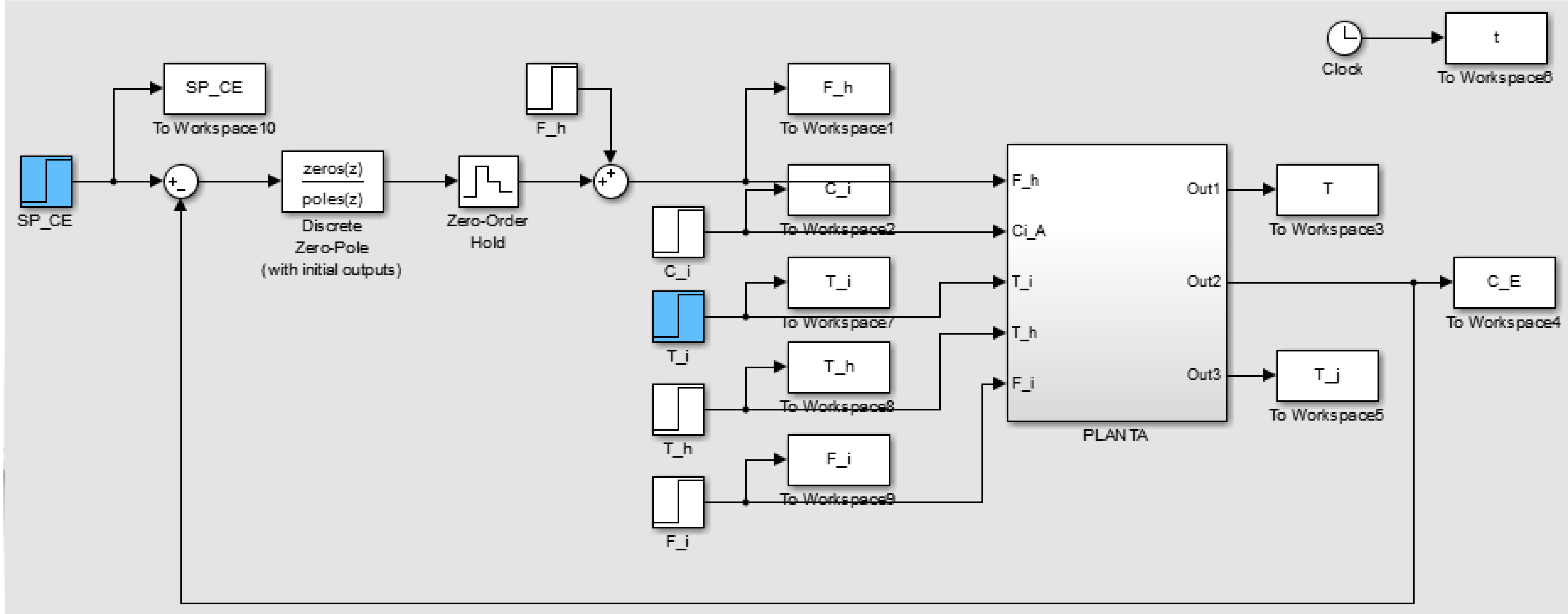
- 3. Estabilidad
  - Dado que $D(z) = \dfrac{1}{G(z)} \dfrac{H(z)}{1 - H(z)}$ entonces:
  - [1–H(z)] debe tener como ceros, todos los polos G(z) que estén fuera de la CU
  - H(z) debe tener como ceros, todos los ceros G(z) que estén fuera de la CU

- 4. Precisión estática
  - (Teor. del val. final) → para tener error de EE nulo para entr. escalón se demuestra:
  - H(z=1) = 1

- Con estas condiciones se plantean ecuaciones para despejar los coeficientes del numerador de H(z)

- Con H(z) y G(z) → computar D(z)

- Cargar "data.mat"
  - Configurar "T" en el bloque controlador y en el z-o-h
- Cargar controlador (formato zpk)
  - Criterio de seguridad para la ganancia
- Analizar discrepancias en los requisitos dinámicos

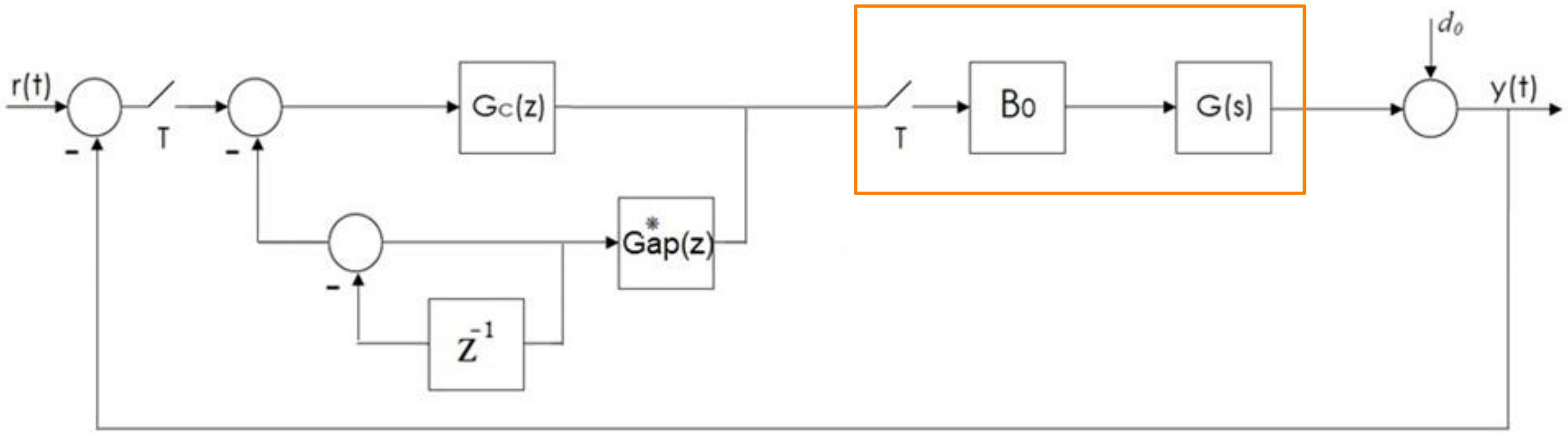


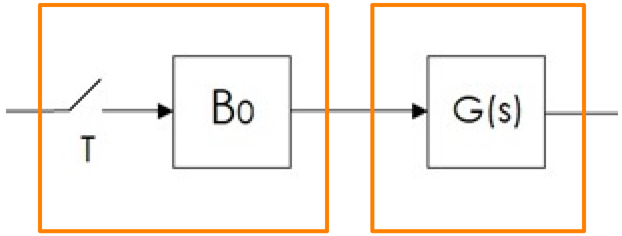
2.2.2. Compensador de Smith

- Compensar el tiempo muerto del proceso

- Varios bloques a diseñar
  - Gc(z) sin considerar el tiempo muerto
  - Modelo del proceso (rectángulo naranja) sin tiempo muerto
  - Tiempo muerto

- Diseño de Gc(z) → 2 caminos posibles





$$\frac{1-e^{-Ts}}{s} \qquad\qquad G^*(s) \quad e^{-\theta_p s}$$

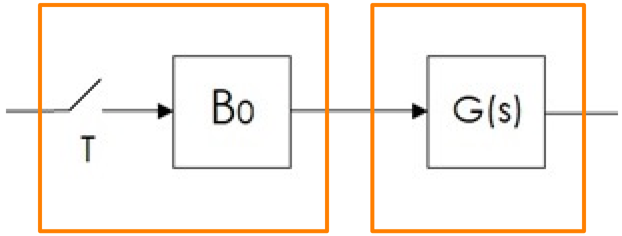
- $G^*_{ap}(s) = \dfrac{1-e^{-Ts}}{s} G^*(s)$ con $e^{-Ts} \cong \dfrac{1-\frac{T}{2}s}{1+\frac{T}{2}s}$ (Padé)

- A partir de $G^*_{ap}(s)$, diseño $G_c(s)$ en Dom. Frecuencial (como en TP2)

- Hallar equivalente discreto $G_c(z)$ con método de mapeo de polos y ceros

- Además $G^*_{ap}(z)$ → equivalente discreto de $G^*(s)$ con mét. Bloq. Equiv.
  → también desde $G^*_{ap}(s)$ con mapeo de polos y ceros

- Implementación: utilizar Simulink Ragazzini como base



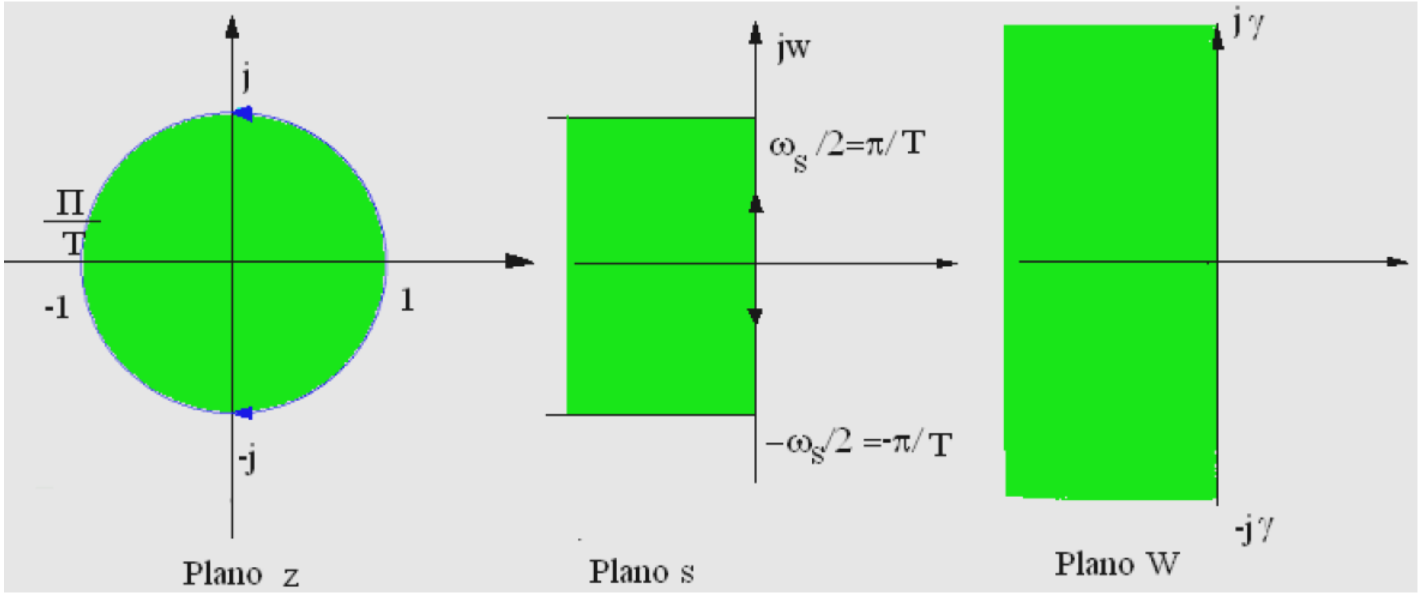
$$\frac{1-e^{-Ts}}{s} \qquad\qquad G^*(s) \quad e^{-\theta_p s}$$

- A partir de $G^*(s)$, hallar $G^*_{ap}(z)$ con mét. Bloq. Equiv.
- Con $G^*_{ap}(z)$ → diseñar $G_c(w)$ a través de Transformada *W*
- Hallar $G_c(z)$ utilizando la antitransformada *W*
- Además $G^*_{ap}(z)$ → equivalente discreto de $G^*(s)$ con mét. Bloq. Equiv.
  → también desde $G^*_{ap}(s)$ con mapeo de polos y ceros
- Implementación: utilizar Simulink Ragazzini como base

Anexo. Diseño de *J(z)* con el método de la Transf. W

- *FT(z)* con $z=e^{jwT}$ no corresponde a una representación racional en w
  - No es posible aplicar métodos de respuesta en frecuencia (TP2)

- Transformada W (bilineal) → W = 2/T * (z-1)/(z+1)
  - Transf de variables para llevar la representación a la forma racional *FT(W)*
  - Poder aplicar procedimiento diseño conocido <u>bajo ciertas hipótesis</u>
  - De teoría: $W = j\,\gamma$ con γ: pseudo-frecuencia en el plano-W
  - Se demuestra que $\gamma \rightarrow w$ cuando $T \rightarrow 0$
  - i.e. podemos trabajar en Dom. W como lo hacemos en Dom. s (diseño en Dom. Frec) siempre que T sea aceptablemente pequeño (*wT/2<<1*)



Anexo. Diseño de *J(z)* vía Transf. W: procedimiento

- 1. *G(s)* (zpk) → método z.o.h. → *G(z)*
- 2. *G(z)* (zpk) → Transf. W → *G(W)*
  - `Gw = transfw(Gz)`
  - var. "s" como var. "w"
  - Al comparar *G(W)* contra *G(s)*:
    - Sus ganancias estáticas coinciden → *G(s=0) = G(W=0)*
    - denominadores tienen = forma → ubicación polos difieren ligeramente (dep. de T)
    - lo mismo ocurre con los ceros
    - aparecen 2 ceros en *G(W)* (por el proceso de muestreo/retención)
      - 1 cero no-mínima fase en W=2/T
      - 1 cero negativo en alta frecuencia
- 3. Eliminar cero de alta frecuencia (*G'(W)*) y corregir la ganancia (*G''(W)*)
  - *K * G'(W=0) = G(s=0)*
- 4. A partir de *G''(W)* diseñar el controlador $G_c(W)$ con el mét. frecuencial (TP2)
  - Verificar que T sea aceptablemente pequeño (*wT/2<<1*)
- 5. $G_c(W)$ → Anti-Transf. W → $G_c(z)$
  - `Gcz = antitransfw(Gcw, T)`