NUMERICAL SOLUTION OF *ODEs* OR *STATE EQUATION SYSTEMS*.

EULER'S METHOD.

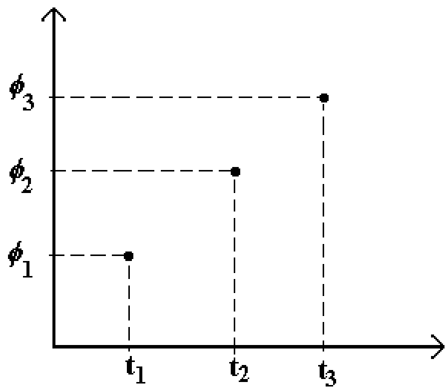
For simplicity, let's consider the scalar case (only one state variable) of the initial value problem:

$$\frac{dx(t)}{dt} = f\left[x(t), t\right] \qquad (1)$$
$$x(t_0) = x_0$$

A numerical approach to the problem of finding a solution to the previous problem demands the discrete representation of time through a sequence of points t_k, $k = 1,2,....,n$. Even if it not necessary and sometimes it could be inconvenient, let's assume to the effect of this introduction, that the points are equally spaced in time: $t_k = k \cdot h$

Let's now use the following notation for the exact solution of the problem:

$f(t = k \cdot h) = f(t_k) = f_k$, where ϕ_k stands for the value of the solution $f(t)$ at instant t_k. A set of pairs (f_k, t_k) will constitute the ***exact*** discrete-time representation of the solution.



Numerical representation of the exact solution.

A numerical method will be next presented yielding a sequence of values x_k constituting a good approximation of the exact values f_k. A set of pairs (x_k, t_k) will constitute the ***approximate*** discrete-time representation of the solution.

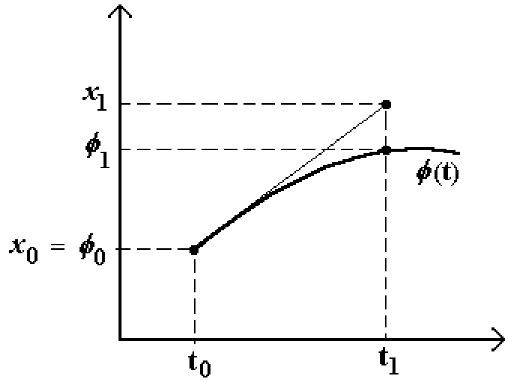
Euler's Method

There are a lot of methods allowing for the obtention of the numerical approximate solution of our problem. Numerical methods for solving differential equations are known as *integration methods*. The most simple and well known is Euler's Method.

Euler's Method proceeds approximating the time derivative of $x(t)$ through the incremental quotient as follows:

$$X(t_k + \Delta t) = X(t_k) + f(X(t_k), t_k) \cdot \Delta t \quad \Rightarrow \quad X_{k+1} = X_k + f_k \cdot h \tag{2}$$

where $\Delta t = h$ is called the *integration step*. See the following graphical interpretation of Euler's formula:



Graphical interpretation of Euler's formula.

Given the initial value problem (1) and Euler's formula (2) the approximate values $X_1, X_2, ...$ can be computed in a sequential way (first compute X_1 using the known value X_0, then use X_1 to calculate X_2, and so on).

Errors: the error due to the numerical approximation is called the ***truncation error***. Even if using an ideal computer this error will be present. The error due to the finite precision representation of numbers in real computers is called the ***round-off error***. It represents an additional source of errors.

Example:

Consider the following first order system:

$\dot{x} = -3x^3 + t$

$x(0) = 1$

Euler's method yields:

$x_{k+1} = x_k + (-3x_k^3 + t_k) \cdot h$,

whose succesive application produces

$x_1 = 1 - 3h \quad x_2 = x_1 + (-3x_1^3 + h) \cdot h \quad x_3 = x_2 + (-3x_2^3 + 2h) \cdot h$, and so on.

EXERCISES

1 – Find the explicit expression for x_2 above and use it to calculate the explicit expression for the next value x_3.

2 – Determine the numerical approximation (i.e., the particular expression for formula (2) above) of the second order State Equation System of the *mass-spring-damper* example via Euler's method. Apply the rule (2) to each of both state equations.

3 – The nonlinear second order SES constitutes a posible version of the famous Lotka-Volterra's model. The variables x_1 and x_2 represent respectively the population of Preys and Predators in a common habitat. This is a simple ***nonlinear*** case where the general solution cannot be analytically obtained. So, the numerical approach is a must !

$$\begin{aligned} \dot{x}_1 &= \epsilon x_1 + \alpha x_1 x_2 - \sigma x_1^2 \\ \dot{x}_2 &= -m x_2 + \beta x_1 x_2 \end{aligned}$$

Apply the rule (2) of Euler's method to each of both state equations. Then particularize the result for the following set of parameter:

$\epsilon = 0.1,\ \alpha = 0.01,\ \sigma = 0.01,\ m = 0.4,\ \beta = 0.5$

Find numerically the three equilibrium points. If you like, write a program to implement the recursive algorithm given by Euler's method and draw the solutions in the x_1-x_2 plane for some set of initial value pairs (x_{10}, x_{20}).
Recall the restriction of the solutions to the first quadrant.

MORE ON NUMERICAL METHODS (Euler Method)

Forward or Explicit Euler

The technique previously described is known as forward or explicit Euler. As already seen, it is based on the following approximation of the time derivative of *x(t)*, which corresponds to the so-called ***forward*** *incremental quotient* (see Fig. "Graphical interpretation of Euler's Formula"):

$$\left.\frac{dx(t)}{dt}\right|_{t=t_k} \approx \left.\frac{\Delta x(t)}{\Delta t}\right|_{t=t_k} = \frac{x(t_k + h_k) - x(t_k)}{h_k} = \frac{x(t_{k+1}) - x(t_k)}{h_k} = \frac{x_{k+1} - x_k}{h_k}$$

As a result, when applied to the dynamical model

$$\boxed{\frac{d\ x(t)}{d\ t} = f\ \big(\ x(t), u(t), t\big)}$$

it yields the approximation

$$x_{k+1} \approx x_k + h_k \,.\, f\ (x_k\ , u_k\ , t_k)$$

which is handled as the identity

$$\boxed{x_{k+1} = x_k + h_k \,.\, f\ (x_k\ , u_k\ , t_k)}$$

in order to compute the numerical approximation to the solution of the differential equation.

The latter formula ***explicitely*** calculates the actualization of the state vector as a function of known values.

Backward or Implicit Euler

In this case, the so-called ***backward*** *incremental quotient* is used in order to approximate the time derivative of *x(t)*:

$$\left.\frac{dx(t)}{dt}\right|_{t=t_k} \approx \left.\frac{\Delta x(t)}{\Delta t}\right|_{t=t_k} = \frac{x(t_k) - x(t_k - h_{k-1})}{h_{k-1}} = \frac{x(t_k) - x(t_{k-1})}{h_{k-1}} = \frac{x_k - x_{k-1}}{h_{k-1}}$$

As a result, when applied to the dynamical model

$$\boxed{\frac{d\ x(t)}{d\ t} = f\ \big(\ x(t), u(t), t\big)}$$

it yields the approximation

$$x_k \approx x_{k-1} + h_{k-1} \,.\, f\ (x_k\ , u_k\ , t_k)$$

which, incrementing the time index in one unit is shown to be equivalent to

$$x_{k+1} \approx x_k + h_k \,.\, f\ (x_{k+1}\ , u_{k+1}\ , t_{k+1})$$

which is handled as the identity

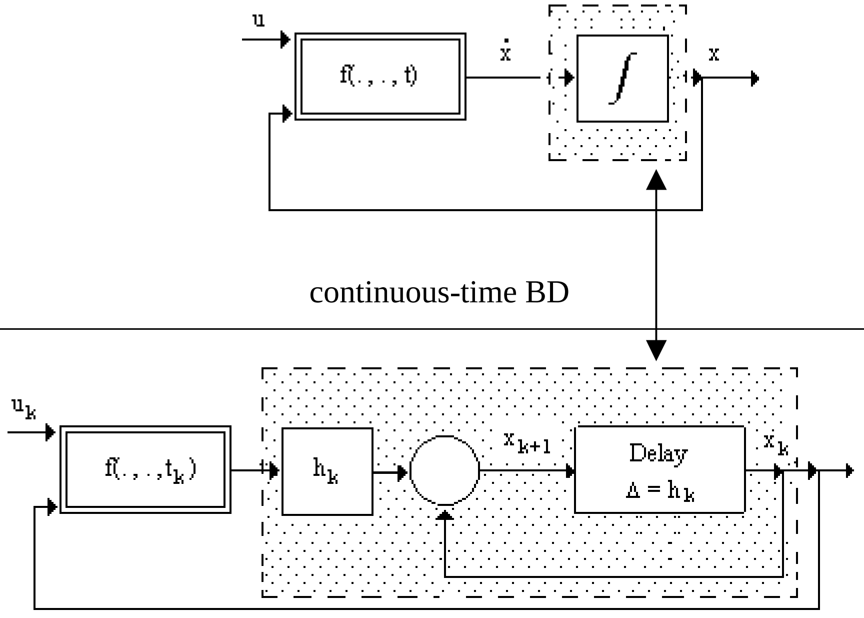
$$\boxed{x_{k+1} = x_k + h_k \,.\, f\ (x_{k+1}\ , u_{k+1}\ , t_{k+1})}$$

The latter formula ***implicitely*** defines the actualization of the state vector, because the right-hand side contains the unknown value x_{k+1} of the state vector and not –as in the previous case– only known variable values (u_{k+1} , t_{k+1}). Thus, the unknown x_{k+1} cannot in general be calculated through a direct evaluation of the right-hand side, but it should be determined with the help of some implicit method.

BLOCK DIAGRAM (BD) REPRESENTATION OF BOTH FORMULAE, FORWARD AND BACKWARD EULER.

Forward Euler

$$\frac{d\ x(t)}{d\ t} = f\ \left(\ x(t), u(t), t\right)$$



$$x_{k+1} = x_k + h_k \ .\ f\ \left(x_k\ , u_k\ , t_k\right)$$

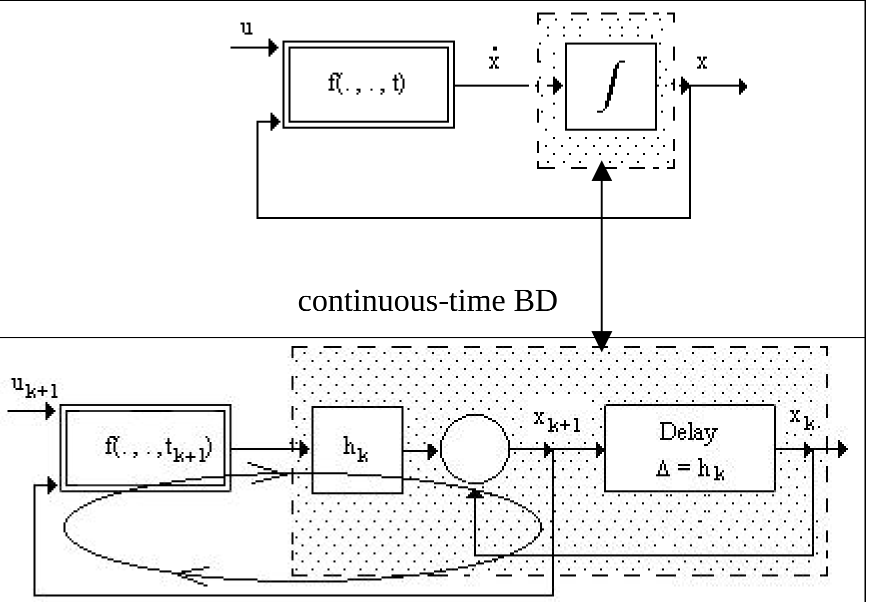
Forward Euler

Fig. FE-BD**:** Forward-Euler discrete-time BD

As shown in the figures above, the method Forward Euler assigns the *subsystem containing the discrete-time delay* as the numerical approximation to the *continuous-time integrator*.

Backward Euler

$$\frac{d\ x(t)}{d\ t} = f\ \left(\ x(t), u(t), t\right)$$



$$x_{k+1} = x_k + h_k \ .\ f\ \left(x_{k+1}\ , u_{k+1}\ , t_{k+1}\right)$$

Backward Euler

Fig. BE-BD**:** Backward-Euler discrete-time BD

As shown in the new set of figures, when using the method Backward Euler, a different discrete-time BD corresponds to the *continuous-time integrator*. Observe the *algebraic loop* in the BD, which is the *graphical expression of an implicit equation*. An algebraic loop is a signal path without dynamical components building a closed loop in a BD, i.e., it has no integrators (or continuous-time delays) in the case of a continuous-time BD, and no discrete-time delays in a discrete-time BD.

N.B. 1: In this case, the algebraic loop (implicit equation) is a consequence of the numerical approximation method used (implicit Euler). It does not exist in the original state-equation system.

N.B. 2: Recall that there exist *continuous* state-equation systems with implicit equations. This is the case for instance of Differential-Algebraic Systems, which when put into the *continuous* BD form will contain algebraic loops due to the algebraic equations.

EXERCISES

First Exercise. Given the continuous-time model

$$\dot{x}(t) = a\,x(t) + b\,u(t) \qquad \text{(scalar variables and coefficients !)}$$

a. Obtain both explicit and implicit discrete-time approximation after forward- and backward-Euler, respectively.
b. The original continuous-problem being linear, it is possible to solve the implicit equation for x_{k+1} , and in this way, to convert the implicit problem into an explicit one. Obtain the explicit solution for x_{k+1} , and analize on it the stability inherent to the backward-Euler method for the free system, i.e., for $u(t) = 0$. (Stability means that if the solution converges for $t \rightarrow \infty$ –as it is the case for $a < 0$– , then, the approximate solution converges for $k \rightarrow \infty$. In general, the stability of a numerical method will depend on the choice of h).
c. Draw the block diagram version of the three previous results.

Second Exercise. Given the continuous-time (CT) model

$$\ddot{y}(t) + a_1\,\dot{y}(t) + a_2\,y(t) = b\,u(t) \qquad \text{(scalar variables and coefficients !)}$$

a. Obtain both explicit and implicit discrete-time (DT) approximation after forward- and backward-Euler, respectively.

Help: a possible technique to solve this problem consists in converting the second order differential equation into a system of two state equations, which are to be discretized later (for instance, with the definitions $x_1 = y$, $x_2 = y\text{-}dot$).

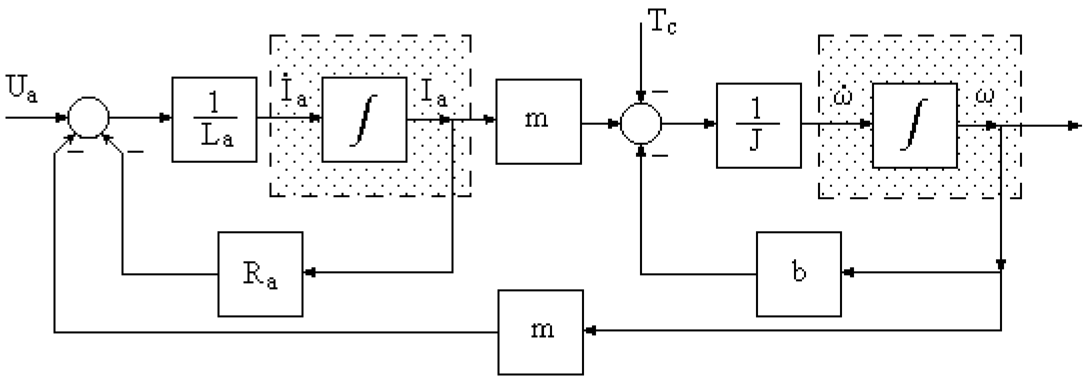
b. Draw the block diagram version of the two previous results.

CT-BD → DT-BD → DT-EQUATIONS

Discretizing directly on the block diagrams, then obtaining the DT-equations:

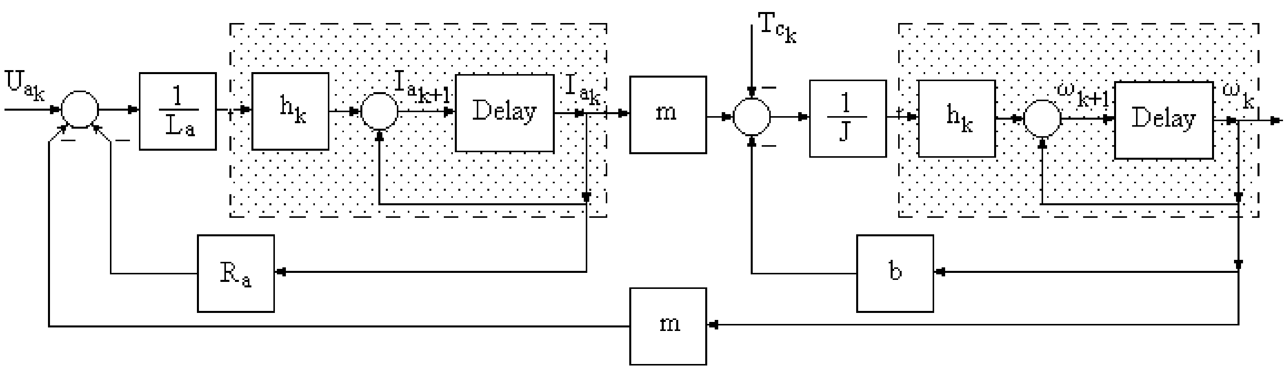
Example: ***PMDCM*** (Permanent Magnet DC Motor)

i) CT Block diagram



ii) Forward Euler DT-BD *is obtained after the BD in* ***Fig. FE-BD*** *above*:

Forward Euler DT-BD



The DT ***Explicit*** State Equations can be directly read from the previous BD as follows:

$$\begin{cases} I_{ak+1} = I_{ak} - h_k \dfrac{R_a}{L_a} I_{ak} - h_k \dfrac{m}{L_a} w_k + h_k \dfrac{1}{L_a} U_{ak} \\ w_{k+1} = w_k - h_k \dfrac{b}{J} w_k + h_k \dfrac{m}{J} I_{ak} - h_k \dfrac{1}{J} T_{ck} \end{cases}$$

or, in matrix form:

$$\begin{bmatrix} I_{ak+1} \\ w_{k+1} \end{bmatrix} = \begin{bmatrix} 1 - h_k \dfrac{R_a}{L_a} & -h_k \dfrac{m}{L_a} \\ h_k \dfrac{m}{J} & 1 - h_k \dfrac{b}{J} \end{bmatrix} \begin{bmatrix} I_{ak} \\ w_k \end{bmatrix} + \begin{bmatrix} h_k \dfrac{1}{L_a} & 0 \\ 0 & -h_k \dfrac{1}{J} \end{bmatrix} \begin{bmatrix} U_{ak} \\ T_{ck} \end{bmatrix}$$

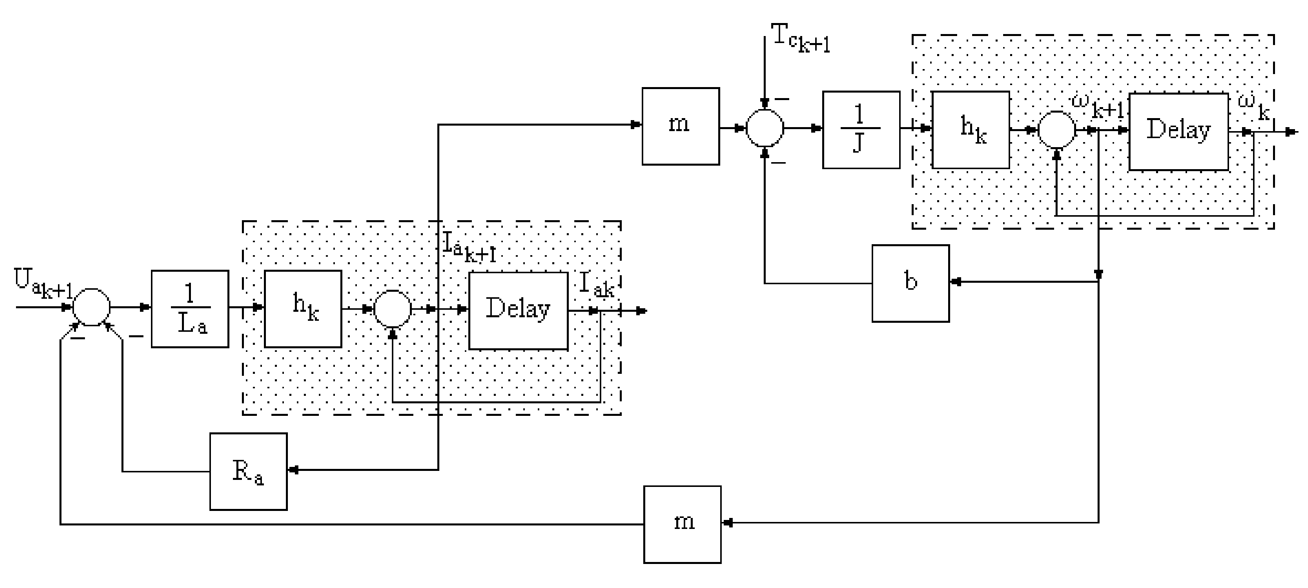
$$\underbrace{\begin{bmatrix} I_{ak+1} \\ \omega_{k+1} \end{bmatrix}}_{X_{k+1}} = \underbrace{\begin{bmatrix} 1 - h_k \dfrac{R_a}{L_a} & -h_k \dfrac{m}{L_a} \\ h_k \dfrac{m}{J} & 1 - h_k \dfrac{b}{J} \end{bmatrix}}_{A_k = A(h_k)} \underbrace{\begin{bmatrix} I_{ak} \\ \omega_k \end{bmatrix}}_{X_k} + \underbrace{\begin{bmatrix} h_k \dfrac{1}{L_a} & 0 \\ 0 & -h_k \dfrac{1}{J} \end{bmatrix}}_{B_k = B(h_k)} \underbrace{\begin{bmatrix} U_{ak} \\ T_{ck} \end{bmatrix}}_{U_k}$$

It can be seen that the DT ***Explicit*** State Equations are of the general form:

$$X_{k+1} = A_k X_k + B_k U_k$$

iii) ***Backward Euler DT-BD*** *is obtained after the BD in Fig. BE-BD above*:

Backward Euler DT-BD



The DT ***Implicit*** State Equations can be directly read from the previous BD as follows:

$$\begin{cases} I_{ak+1} = I_{ak} - h_k \dfrac{R_a}{L_a} I_{ak+1} - h_k \dfrac{m}{L_a} w_{k+1} + h_k \dfrac{1}{L_a} U_{ak+1} \\ w_{k+1} = w_k - h_k \dfrac{b}{J} w_{k+1} + h_k \dfrac{m}{J} I_{ak+1} - h_k \dfrac{1}{J} T_{ck+1} \end{cases}$$

or, in matrix ***Implicit*** form:

$$\begin{bmatrix} I_{ak+1} \\ w_{k+1} \end{bmatrix} = \begin{bmatrix} I_{ak} \\ w_k \end{bmatrix} + \begin{bmatrix} -h_k \dfrac{R_a}{L_a} & -h_k \dfrac{m}{L_a} \\ h_k \dfrac{m}{J} & -h_k \dfrac{b}{J} \end{bmatrix} \begin{bmatrix} I_{ak+1} \\ w_{k+1} \end{bmatrix} + \begin{bmatrix} h_k \dfrac{1}{L_a} & 0 \\ 0 & -h_k \dfrac{1}{J} \end{bmatrix} \begin{bmatrix} U_{ak+1} \\ T_{ck+1} \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} I_{ak+1} \\ \omega_{k+1} \end{bmatrix}}_{X_{k+1}} = \underbrace{\begin{bmatrix} I_{ak} \\ \omega_k \end{bmatrix}}_{X_k} + \underbrace{\begin{bmatrix} -h_k \dfrac{R_a}{L_a} & -h_k \dfrac{m}{L_a} \\ h_k \dfrac{m}{J} & -h_k \dfrac{b}{J} \end{bmatrix}}_{A_k = A(h_k)} \underbrace{\begin{bmatrix} I_{ak+1} \\ \omega_{k+1} \end{bmatrix}}_{X_{k+1}} + \underbrace{\begin{bmatrix} h_k \dfrac{1}{L_a} & 0 \\ 0 & -h_k \dfrac{1}{J} \end{bmatrix}}_{B_k = B(h_k)} \underbrace{\begin{bmatrix} U_{ak+1} \\ T_{ck+1} \end{bmatrix}}_{U_{k+1}}$$

$$X_{k+1} = X_k + A_k X_{k+1} + B_k U_{k+1}$$

As the model is linear, an explicit expression can be recovered, as follows:

$$\begin{bmatrix} I_{ak+1} \\ w_{k+1} \end{bmatrix} = \begin{bmatrix} 1 - h_k \dfrac{R_a}{L_a} & -h_k \dfrac{m}{L_a} \\ h_k \dfrac{m}{J} & 1 - h_k \dfrac{b}{J} \end{bmatrix}^{-1} \left(\begin{bmatrix} I_{ak} \\ w_k \end{bmatrix} + \begin{bmatrix} h_k \dfrac{1}{L_a} & 0 \\ 0 & -h_k \dfrac{1}{J} \end{bmatrix} \begin{bmatrix} U_{ak+1} \\ T_{ck+1} \end{bmatrix} \right)$$

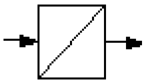
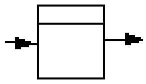
$$X_{k+1} = (I - A_k)^{-1} (X_k + B_k U_{k+1})$$

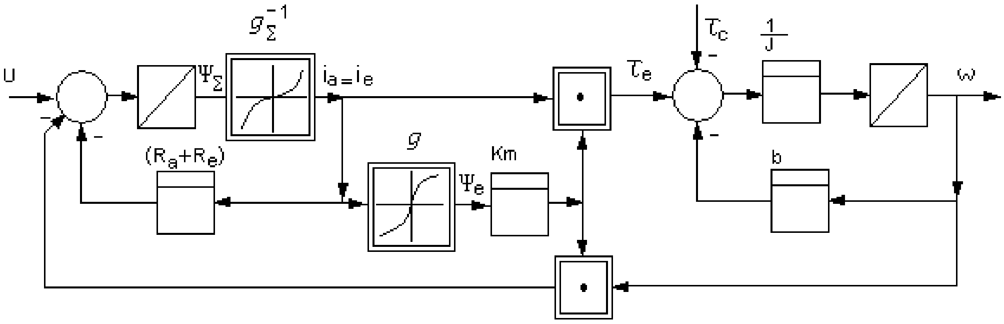
Third Exercise. Consider the previously handled Lotka-Volterra (non-linear) model. Obtain the DT State Equations following the method in the preceding example, id est:

a. Construct the corresponding CT-BD.
b. Construct both, the explicit and the implicit DT-BD´s.
c. Write-down the DT State Equations through reading of the BD´s.

Fourth Exercise. The following is the (non-linear) CT-BD of a Series Connected DC-Motor with full excitation(*). Do the same exercise as in both previous cases. Consider g and g_S as known non-linear functions, and g_S^{-1} as the inverse of the latter.

Meaning of the symbols in the BD :

This block is an *integrator*:	and this one is a *gain*, the value of its gain being Km:
	



(*) Just for information, find below the equivalent circuit of the DC-Motor (if you are not interested, ignore it). *g* is a non-linear function representing the dependence of the magnetic excitation flux y_e on the excitation current *Ie* : $\varnothing_e = g(I_e)$. In a full series connection of both the armature and the field coils, the armature and the excitation currents are the same: *Ia* = *Ie*. This situation is modeled as having a unique coil having $g_\Sigma(I_a) = \varnothing_e + \varnothing_a =$ $g(I_e = I_a) + L_a I_a$ as its magnetics characteristic.