

Basic concepts for using the POG Modeling Program

The POG Modeling Program can be used only if the following basic concepts of the POG modeling technique are known.

Energetic domains:

- Electrical (capacitors, inductors, resistances)
- Mechanical translational (masses, springs, dampers)
- Mechanical rotational (inertias, rot. springs, rot. dampers)
- Hydraulic (hyd. capacitors, hyd. inductors, hyd. resistances)

Each energetic domain is characterized by:

- *2 dynamic elements* “ \mathcal{D}_e ” and “ \mathcal{D}_f ” which store the energy (i.e. capacitors, inductors, masses, springs, etc.);
- *1 static element* “ \mathcal{R} ” which dissipates (or generates) the energy (i.e. resistors, frictions, etc.);

The dynamics of physical systems can be described using 4 variables:

- *2 energy variables* q_e and q_f which define *how much energy is stored* within the *dynamic elements*;
- *2 power variables* v_e and v_f which describe *how the energy moves* within the system.

The *energy variables* are the time integrals of the *power variables*.

$$q_e = \int_0^t v_f dt \qquad q_f = \int_0^t v_e dt$$

Dynamic structure of the energetic domains:

	Electrical	Mech. Tras.	Mech. Rot.	Hydraulic
\mathcal{D}_e	C Capacitor	M Mass	J Inertia	C_I Hyd. Capacitor
q_e	Q Charge	p Momentum	p Ang. Momentum	V Volume
v_e	V Voltage	\dot{x} Velocity	ω Ang. Velocity	P Pressure
\mathcal{D}_f	L Inductor	E Spring	E Spring	L_I Hyd. Inductor
q_f	ϕ Flux	x Displacement	θ Ang. Displacement	ϕ_I Hyd. Flux
v_f	I Current	F Force	τ Torque	Q Volume flow rate
\mathcal{R}	R Resistor	b Friction	b Ang. Friction	R_I Hyd. Resistor

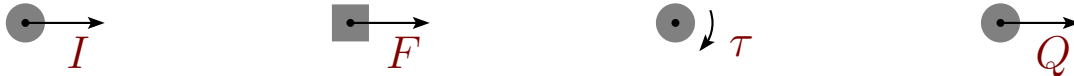
Each energetic domain is characterized by two power variables of different type: an **effort/across-variable** and a **flow/through-variable**.

POG variables	Electrical	Mech. Tras.	Mech. Rot.	Hydraulic
Across-var.: v_e	V Voltage	\dot{x} Velocity	ω Angular vel.	P Pressure
Through-var.: v_f	I Current	F Force	τ Torque	Q Volume flow rate

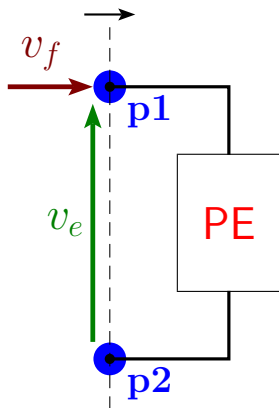
The “**effort/across-variables**” (voltage V , velocity \dot{x} , angular velocity ω and pressure P) are defined “*between two points*” of the space:



The “**flow/through-variables**” (current I , force F , torque τ and volume flow rate Q) are defined “*in each point*” of the space:



Each Physical Element “**PE**”:



1. interacts with the external world by means (in the simplest case) of two terminals **p1** and **p2**;
2. the energy is stored or dissipated (converted into heat) “within” the physical element;
3. the energy enters or exits the physical element only using the two power variables v_e and v_f .

Assumptions on the power variables v_e and v_f :

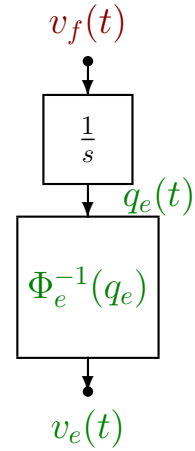
- The **effort/across-variable** v_e is positive if defined between terminals **p1** (top) and **p2** (bottom): $v_e = v_{p1} - v_{p2}$;
- The **flow/through-variable** v_f is positive if it enters terminal **p1** and exits from terminal **p2**;
- The power is positive if it enters the Physical element “**PE**”.

Each *dynamic effort block* \mathcal{D}_e is characterized by:

- 1) a *power through-variable* $v_f(t)$ as input;
- 2) an internal *energy across-variable* $q_e(t)$:

$$q_e(t) = \int_0^t v_f(t) dt$$

- 3) a *power across-variable* $v_e(t)$ as output;
- 4) a *constitutive relation* $q_e = \Phi_e(v_e)$ which links the internal energy variable $q_e(t)$ to the output power variable $v_e(t)$;



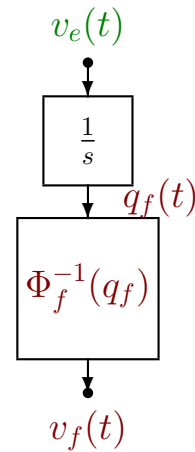
The *differential equation* of the *dynamic element* \mathcal{D}_e is: $\dot{q}_e(t) = v_f(t)$.

Each *dynamic flow block* \mathcal{D}_f is characterized by the corresponding “dual structure”:

- 1) a *power across-variable* $v_e(t)$ as input;
- 2) an internal *energy through-variable* $q_f(t)$:

$$q_f(t) = \int_0^t v_e(t) dt$$

- 3) a *power through-variable* $v_f(t)$ as output;
- 4) a *constitutive relation* $q_f = \Phi_f(v_f)$ which links the internal energy variable $q_f(t)$ to the output power variable $v_f(t)$;



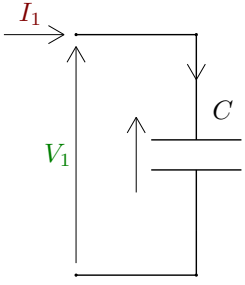
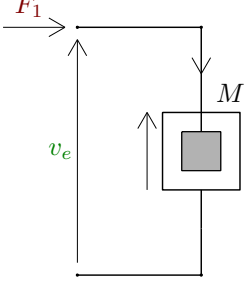
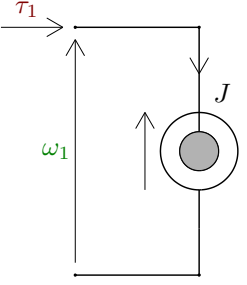
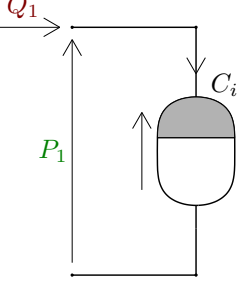
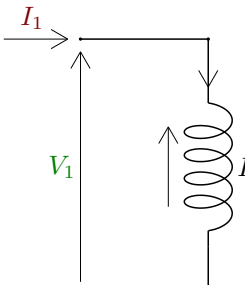
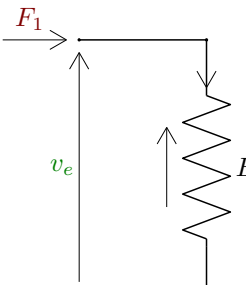
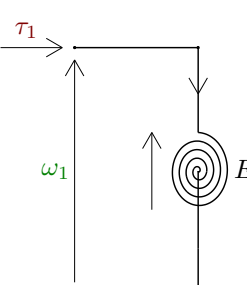
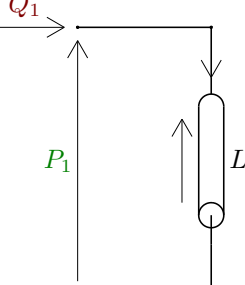
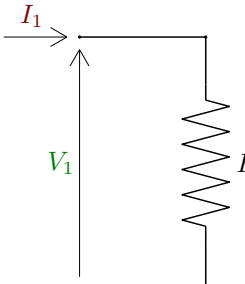
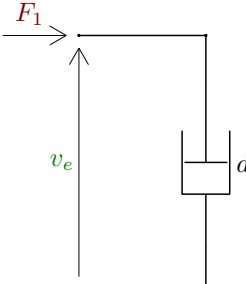
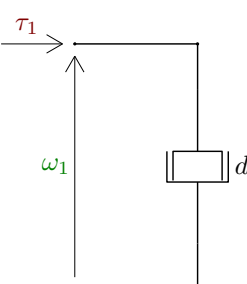
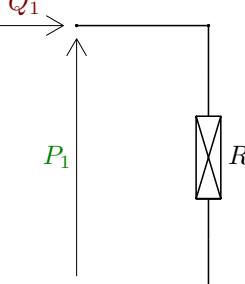
The *differential equation* of the *dynamic element* \mathcal{D}_f is: $\dot{q}_f(t) = v_e(t)$.

In the POG modeling technique, all the Physical Elements “PE” are identified by using a two digit string “xX”:

- Electrical: (eC, eL, eR) and (eV, eI).
- Mechanical translational: (mM, mK, mB) and (mV, mF).
- Mechanical rotational: (rJ, rK, rB) and (rW, rT).
- Hydraulic: (iC, iL, iR) and (iP, iQ).

There are two different types of Physical Elements: the “*Internal Physical Elements*” and the “*External Physical Elements*”.

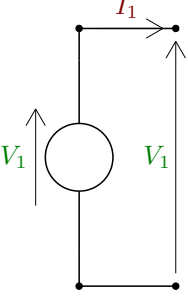
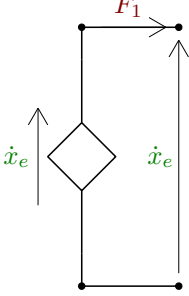
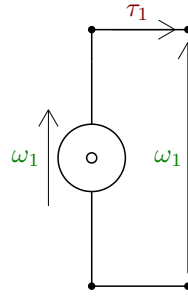
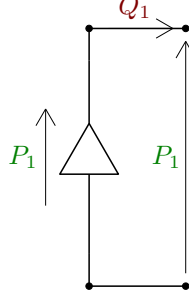
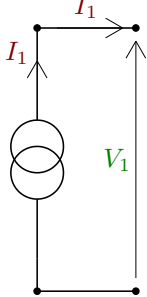
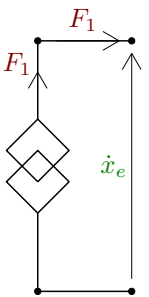
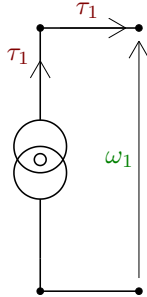
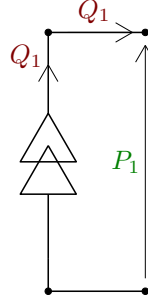
Two digit string “**xX**” of the “*Internal Physical Elements*” “**PE**”:

	Electrical	Mech. Tras.	Mech. Rot.	Hydraulic
Element \mathcal{D}_e	<p>eC</p>  <p>Capacitor</p>	<p>mM</p>  <p>Mass</p>	<p>rJ</p>  <p>Inertia</p>	<p>iC (iK)</p>  <p>Hyd. Capacitor</p>
Element \mathcal{D}_f	<p>eL</p>  <p>Inductor</p>	<p>mE (mK)</p>  <p>Spring</p>	<p>rE (rK)</p>  <p>Rot. Spring</p>	<p>iL</p>  <p>Hyd. Inductor</p>
Element \mathcal{R}	<p>eR (eG)</p>  <p>Resistor</p>	<p>mD (mB)</p>  <p>Friction</p>	<p>rD (rB)</p>  <p>Ang. Friction</p>	<p>iR (iG)</p>  <p>Hyd. Resistor</p>

The “*Internal Physical Elements*” are the elements that belong to the considered system and describe how the system stores and dissipates energy. For all these *Elements* the Power is positive if it enters into the element.

For each Energetic Domain there are only three “*Internal Elements*” : the two dynamic elements \mathcal{D}_e and \mathcal{D}_f , which store energy, and the static element \mathcal{R} , which dissipates energy.

Two digit string “**xX**” of the “*External Physical Elements*” “**PE**”:

	Eletttric	Mecc. Tras.	Mecc. Rot.	Hydraulic
Across-generators \mathcal{G}_e	<p>eV</p>  <p>Voltage Gen.</p>	<p>mV</p>  <p>Velocity Gen.</p>	<p>rW</p>  <p>Ang. Velocity Gen.</p>	<p>iP</p>  <p>Pressure Gen.</p>
Through-generators \mathcal{G}_f	<p>el</p>  <p>Current Gen.</p>	<p>mF</p>  <p>Force Gen.</p>	<p>rT</p>  <p>Torque Gen.</p>	<p>iQ</p>  <p>Flow Rate Gen.</p>

The “*External Physical Elements*” are the elements that **do not** belong to the considered system, but describe how the system interacts with the external world.

For all the “*External Physical Elements*” the Power is positive if it exits from the element.

For each Energetic Domain there are only two “*External Elements*”:

- 1) an **across-generator \mathcal{G}_e** which generates the **power across-variable v_e** ;
- 2) a **through-generator \mathcal{G}_f** which generates the **power through-variable v_f** .

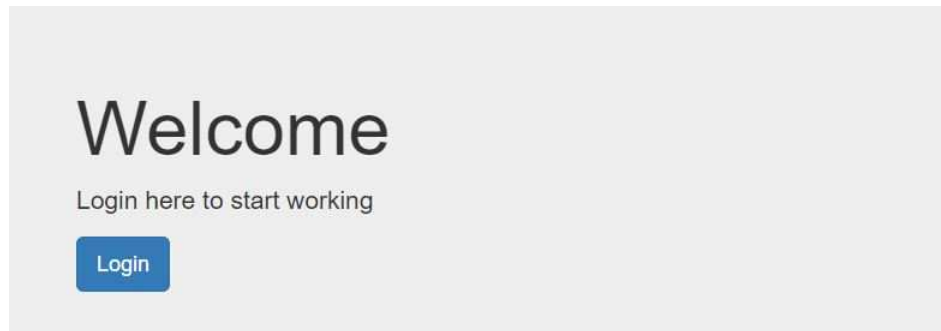
These “*External Elements*” provide the inputs of the considered system. They describe how the external world acts on the considered system.

POG Modeler

To log on to the **POG Modeler** program, use the following link:

<http://zanasi2009.ing.unimo.it>

The Welcome page:



The Sign Up page:

Register

Nome utente*

Obbligatorio. 150 caratteri o meno. Solo lettere, cifre e @/./+/-/_

Email*

indirizzo email

Password*

- La tua password non può essere troppo simile alle altre tue informazioni personali.
- La tua password deve contenere almeno 8 caratteri.
- La tua password non può essere una password comunemente usata.
- La tua password non può essere interamente numerica.

Conferma password*

Inserisci la stessa password inserita sopra, come verifica.

Use the link in the email message:

Dear Surname_Name

Here's your activation key for POG Modeler at `zanasi2009.ing.unimo.it` (155.185.49.0):

`http://155.185.49.0/accounts/activate/Ilphbm5hX3JpYW5jYSI:1e1XEJ:KYkdeRkldUxJ`

You have 4 days to complete your registration

POG Modeler can be used by the UNIMORE students of the course "Modeling and Control of Electromechanical Systems".

Do not reply to this mail.

Sincerely,

Roberto Zanasi

The Login page:

Login

Nome utente*

Surname_Name

Password*

.....

Submit

Don't have an account? [Sign Up Here](#)

Forgot Password? [Reset the password here](#)

The Home page:

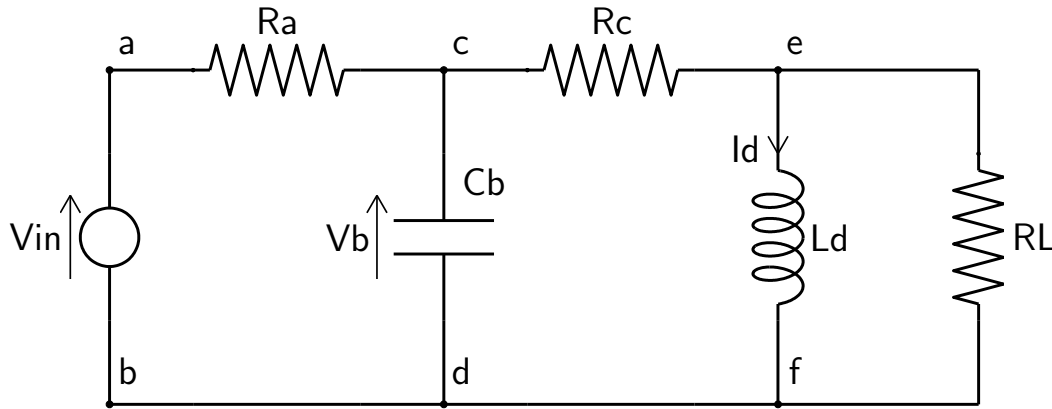
POG Modeler

Help Projects Zanasi

Projects Home

N	Name	Description	Launch Date	Results
1	Circuito_RCLRC	Description	Lun 09/10/17 - 02:47.42	Results Launch Edit Delete
2	Massa_Molla	Description	Lun 09/10/17 - 02:48.30	Results Launch Edit Delete
3	Prova	Description	Lun 09/10/17 - 02:49.29	Results Launch Edit Delete
4	Prova2	Description	Lun 09/10/17 - 02:50.57	Results Launch Edit Delete
5	Uno	Description	Mer 04/10/17 - 05:51.17	Results Launch Edit Delete

Let us consider the following Physical System:



This Physical System belongs to the “Electric” Energetic Domain, but the procedure described in the following slides applies, in the same way, to Physical Systems belonging to other Energetic Domains.

The above Physical System can be introduced in the POG Modeling Program using the following ascii “command-lines”:

```
** , As, No
eV, a, b, An, -90, En, Vin
eR, a, c, Kn, Ra
--, b, d
eC, c, d, En, Vb, Kn, Cb
eR, c, e, Kn, Rc
--, d, f
eL, e, f, Fn, Id, Kn, Ld
eR, e, f, Sh, 0.6, Kn, RL
```

There are two types of command-lines:

- 1) the “*system command-lines*”, such as ‘** , As, No’;
- 2) the “*element command-lines*”, such as ‘eR, a, c, Kn, Ra’.

The “system command-lines” define the parameters that apply to the whole system, that is to all the physical elements belonging to the considered Physical System.

The “element command-lines” introduce new Physical Elements in the POG physical scheme and define their parameters.

Each “system command-line” has the following structure:

Mandatory	Optional
** , Par1 , Val1	[, Par2 , Val2] [, Par3 , Val3] ...

The initial string “******” identifies the line as a “*system command-line*”.

- All the couples ‘, Parx, Valx’ are composed of a system parameter “Parx” and its value “Valx”.
- At least one couple ‘, Par1, Val1’ has to be present in each “*system command-line*”.
- The user can use as many “*system command-lines*” as he wants.
- In the command-line the couples ‘Parx, Valx’ can be given in any order.

Example: the *system command-line* ‘******, As, No’, in the previous example, sets the system parameter “As, No”. This parameter tells the POG Modeling Program: “*not to compute the differential equations of the considered system*”.

Each “element command-line” has the following structure:

Mandatory	Optional
xX , p1 , p2	[, Par1 , Val1] [, Par2 , Val2] ...

The *Mandatory part* of the “*element command-line*” is composed of three elements separated by commas:

- xX** : a two digit string that uniquely identifies the Physical Element;
- p1** : an ascii label identifying the position of the first terminal;
- p2** : an ascii label identifying the position of the second terminal;

The *Optional part* of the “*element command-line*” is composed of “couples” having the following structure: [, **Parx**, **Valx**].

Each “couple” is composed of a parameter “**Parx**” which applies to the Physical Element, and its value “**Valx**”.

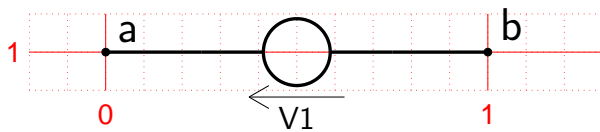
How to insert, step by step, a new System

Let us now describe, step by step, how the given electric scheme can be inserted in the POG Modeling Program.

Note: in the following slides the commands are shown on the left, and the corresponding graphical output is shown on the right.

A voltage generator “eV” can be inserted as follows:

** , As, No, SG, Si
eV, a, b



The first *system command-line* forces the POG Modeling System to:

- 1) not analyze the System (“As, No”);
- 2) show the grid in the plot (“SG, Si”).

Note: the **thin red lines** in the figure are the grid lines of the plot.

The initial string “eV” of the *element command-line* 'eV, a, b' uniquely identifies the electric voltage generator.

The two parameters “a” and “b” are the labels identifying the two terminals **p1** and **p2** of the Physical Element “eV”.

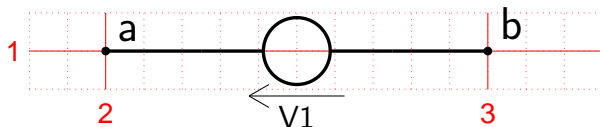
Since the two points “a” and “b” are not defined, the POG Modeling Program sets their positions as follows (default values):

$$“a”=(0,1) \quad \text{and} \quad “b”=“a”+(1,0)=(1,1)$$

Note: the positions of points “a”, “b”, ... of the POG physical scheme will be expressed in the cartesian form: “a”=(0,1), “b”=(1,1), ...

The position of point “a” can be defined as follows:

** , As, No, SG, Si
eV, a=(2+1*1i), b

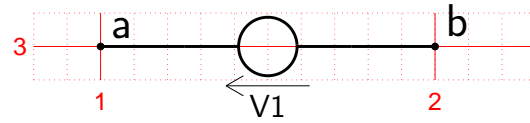


The POG Modeling Program sets the position of point “b” as follows:

$$“a”=(2+1*1i)=(2,1) \quad \rightarrow \quad “b”=“a”+(1,0)=(3,1)$$

Defining the position of a point “b”, one obtains the following result::

** , As, No, SG, Si
eV, a, b=(2+3*1i)



In this case the position of point “a” is chosen as follows:

$$“b”=(2+3*1i)=(2,3) \rightarrow “a”=“b”-(1,0)=(1,3)$$

Note: the positions of points “a”, “b”, ... in the POG scheme are defined using complex numbers. The string ‘b=(2+3*1i)’, for example, sets the position of point “b” as follows: “b”=(2,3). The symbol “1i” in the previous string represents the imaginary unit.

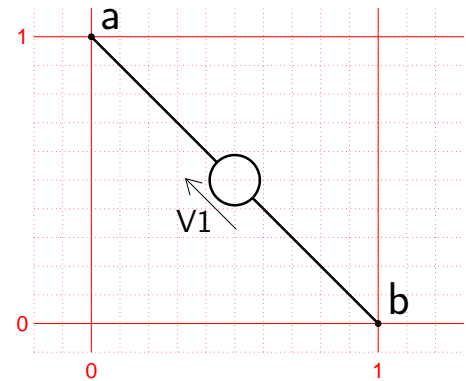
The positions of points “a”, “b”, ... in the POG scheme can be defied in “**absolute**” or “**relative**” way. The following two sets of strings, for example, generate the same result:

1) “**absolute**”:

** , As, No, SG, Si
eV, a=(0+1*1i), b=(1+0*1i)

2) “**relative**”:

** , As, No, SG, Si
eV, a=(0+1*1i), b, An,-45, Ln,sqrt(2)



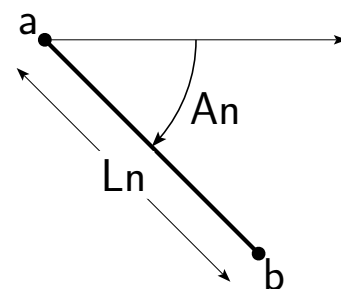
The “**absolute**” way is easy to understand because the exact position of both points “a” and “b” is given.

In the “**relative**” way the new points are defined with respect to the previously defined point using the parameters “An” and “Ln”:

Point “b” is computed as follows:

$$“b” = “a” + Ln * e^{An * 1i}$$

The default values of parameters “An” and “Ln” are: “An, 0” and “Ln, 1”.

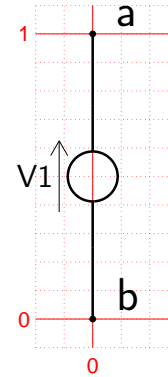


In the above example they are “An, -45°” and “Ln, $\sqrt{2}$ ”.

The voltage generator “eV” can be inserted in vertical positions as follows:

******, As, No, SG, Si

eV, a, b, An, -90



In this case:

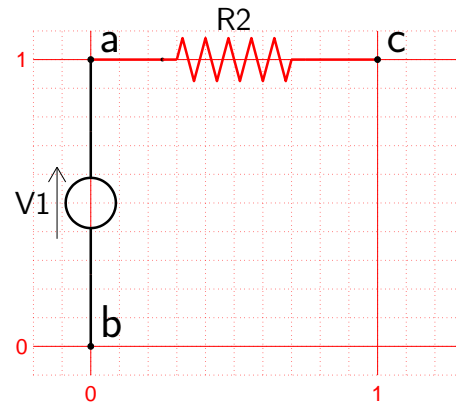
$$“a” = (0,1), \quad “b” = “a” + 1 * e^{-90^\circ * 1i} = (0,0)$$

A resistor “eR” can be added to the scheme as follows (see the last *command-line*):

******, As, No, SG, Si

eV, a, b, An, -90

eR, a, c



The new command-line 'eR, a, c' inserts a resistor “eR” between points “a” and “c”. The position of point “a” is equal to the default value: “a”=(0,1). The position of the new point “c” is defined in a “relative way”, with respect to point “a”, as follows:

$$“c” = “a” + L_n * e^{A_n * 1i} = (0,1) + 1 * e^{0^\circ * 1i} = (1,1)$$

In this case the default values “An, 0” and “Ln, 1” have been used.

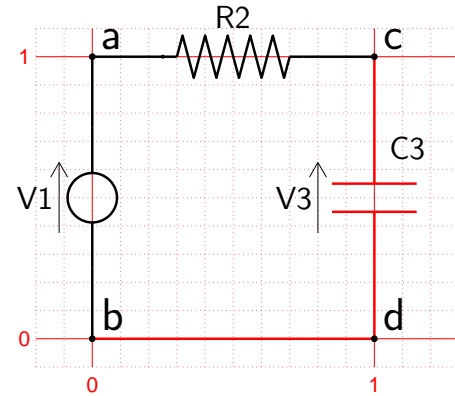
Note: the name “V1” of the input voltage and the name “R2” of the resistance are the “default names” given by the POG Modeling Program. The number that ends the “default names” starts from 1 and is increased by one unit any time a new Physical Element is inserted in the scheme. The number of the default name “V1” is “1” because this was the first element inserted in the scheme; the number of the default name “R2” is “2” because this was the second element inserted in the scheme.

A capacitor “eC” can be added to the scheme as follows (see the last two *command-lines*):

```

**, As, No, SG, Si
eV, a, b, An, -90
eR, a, c
--, b, d
eC, c, d

```



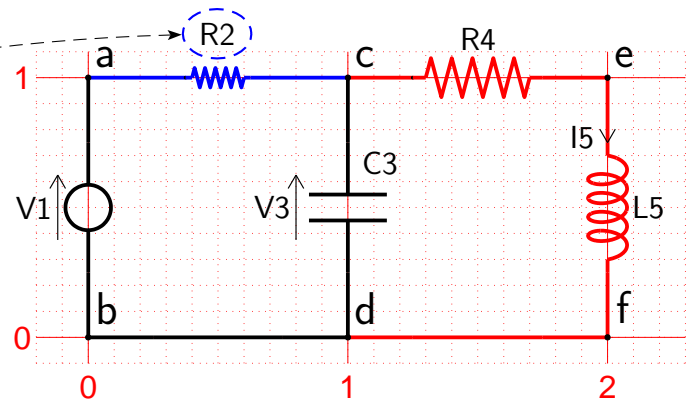
The string “--” in the *command-line* ‘-- , b, d’ tells the POG Modeling Program to insert, in the scheme, a line connecting point “b”=(0,0) to the new point “d” = “b” + (1,0) = (1,0). Then the *command-line* ‘eC, c, d’ inserts a capacitor “eC” between points “c” and “d”.

A new resistor “eR” and a new inductor “eL” can be added to the previous scheme as follows (see the last three *command-lines*):

```

**, As, No, SG, Si
eV, a, b, An, -90
eR, a, c, (Zm, 0.5)
--, b, d
eC, c, d
eR, c, e
--, d, f
eL, e, f

```



The two new points “e” and “f” are defined, using the default values “An, 0” and “Ln, 1”, in a relative way with respect to points “c” and “d”:

$$“e” = “c” + (1,0) = (2,1) \quad \text{and} \quad “f” = “d” + (1,0) = (2,0).$$

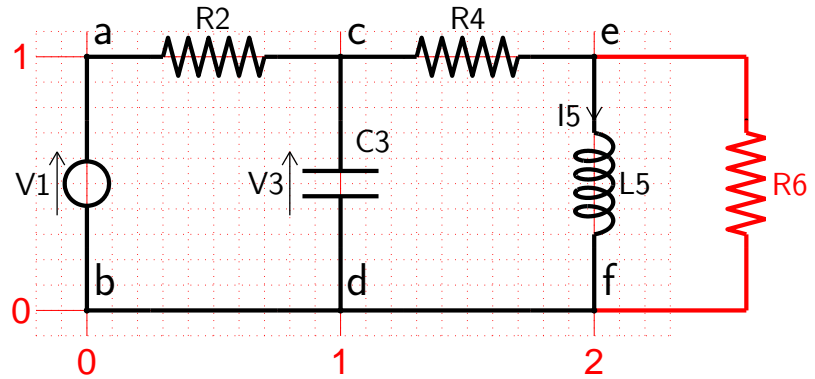
Note: the parameter “Zm, 0.5”, present within the *command-line* ‘eR, a, c, Zm, 0.5’, forces the POG Modeling System to draw the corresponding Physical Element “eR” with a dimension that is 50% smaller than the original one. If parameter “Zm, 1.3” is used, the corresponding Physical Element “eR” will be drawn with a dimension that is 30% larger than the original one.

One last resistor “eR” can be added to the previous scheme as follows (see the last *command-line*):

```

**, As, No, SG, Si
eV, a, b, An, -90
eR, a, c
--, b, d
eC, c, d
eR, c, e
--, d, f
eL, e, f
eR, e, f, Sh, 0.6

```



The new resistor “eR” has been inserted in parallel to inductor “eL”.

The string ‘,Sh, 0.6’, present in the last command-line, tells the POG Modeling Program to shift laterally resistor “eR” by 0.6 units.

Example of use of parameter “Sh”:

```

**, As, No, SG, Si
*,P, e=(2+1*1i)
*,P, f=(2+0*1i)
eR, e, f

```

```

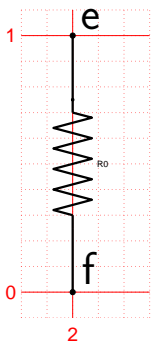
**, As, No, SG, Si
*,P, e=(2+1*1i)
*,P, f=(2+0*1i)
eR, e, f, Sh, 0.6

```

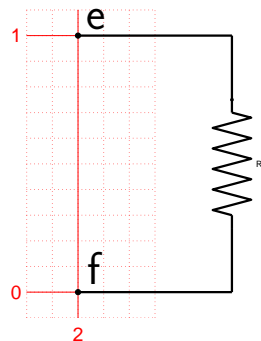
```

**, As, No, SG, Si
*,P, e=(2+1*1i)
*,P, f=(2+0*1i)
eR, e, f, Sh, -0.3

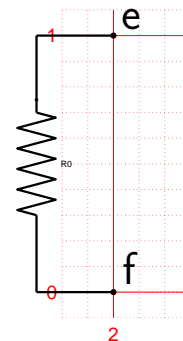
```



⇒



⇒



Note: the value of parameter “Sh” can be either positive or negative.

In the above examples the string ‘*,P, e=(2+1*1i)’ is an example of the following “*special common-line*”:

Mandatory
Optional

$*P, p1 = (x1 + y1 * 1i) \quad [, p2 = (x2 + y2 * 1i)] \dots$

String “*P” identifies the above “*special common-line*”. Each string “pn=(xn+yn*1i)” defines the position of a new point: $pn=(xn, yn)$.

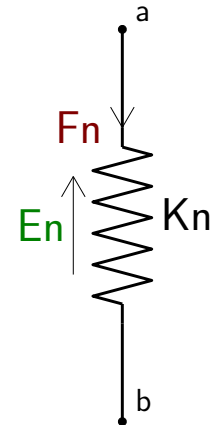
How to change the defaults names

Each Physical Element is characterized by the following variables: an **across-variable** v_e , a **through-variable** v_f and an internal-parameter \mathcal{K} .

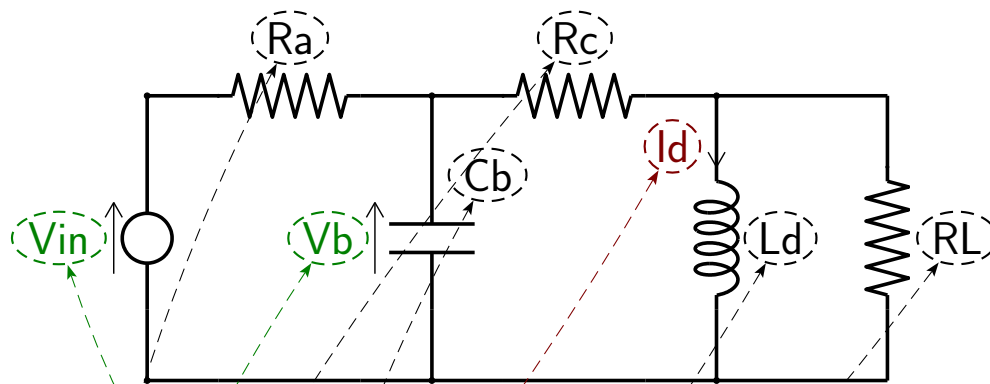
The name of the **across-variable** v_e can be changed using the parameter “En”;

The name of the **through-variable** v_f can be changed using the parameter “Fn”;

The name of the internal-parameter \mathcal{K} can be changed using the parameter “Kn”;



Using the three parameters “En”, “Fn” and “Kn” the user can change the names of all the variables and all the parameters of the considered Physical System:



The final list of command-lines is:

**, As, No, Sn, No

eV, a, b, An, -90, (En, Vin)

eR, a, c, (Kn, Ra)

--, b, d

eC, c, d, (En, Vb), (Kn, Cb)

eR, c, e, (Kn, Rc)

--, d, f

eL, e, f, (Fn, Id), (Kn, Ld)

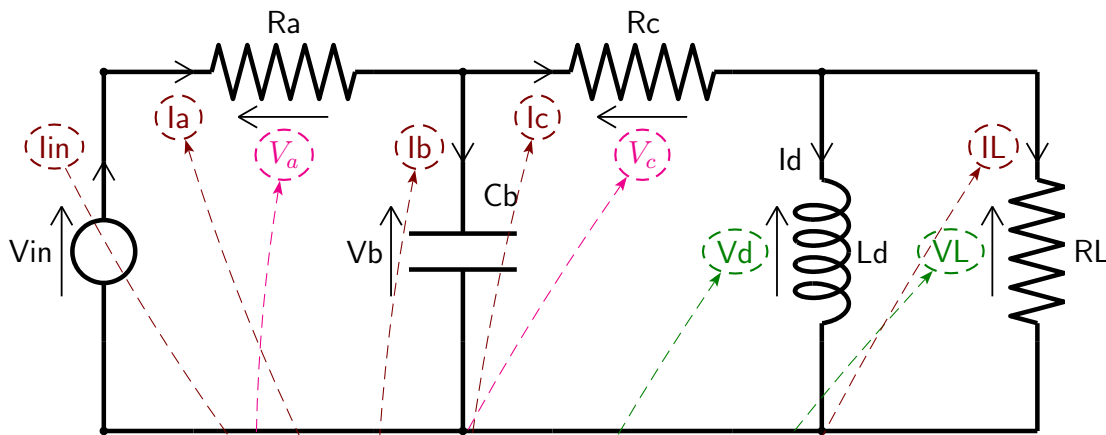
eR, e, f, Sh, 0.6, (Kn, RL)

The three parameters “En”, “Fn” and “Kn” have to be added, in any order, at the end of the line where the Physical Element has been defined.

Default behavior of the POG Modeling Program:

1. the internal parameter “Kn” is shown for all the Physical Elements.
2. for the dynamic elements “ \mathcal{D}_e ” and “ \mathcal{D}_f ”, the Program also shows the “output power variable”, that is the **across-variable** “ v_e ” for the elements “ \mathcal{D}_e ” and the **through-variable** “ v_f ” for the elements “ \mathcal{D}_f ”.
3. for the static elements “ \mathcal{R} ”, only the internal parameter “Kn” is shown.

All the “**input**” and “**output**” power variables are shown if the following general parameters are used: “**ShX, Si**” and “**ShY, Si**”. Adding the *general command-line* ‘******, **ShX, Si, ShY, Si**’ to the previous example, one obtain the following result:

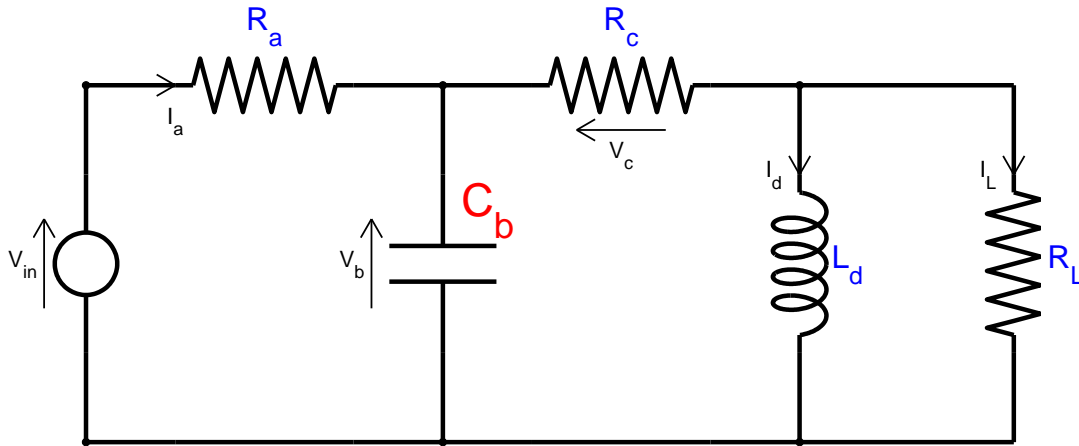


The new shown variables can be properly renamed as follows:

******, As, Si, Sn, No, **ShX, Si**, **ShY, Si**, Us, Si
 eV, a, b, An, -90, En, Vin, (Fn, I_{in})
 eR, a, c, Kn, Ra, (En, V_a) , (Fn, I_a)
 --, b, d
 eC, c, d, En, Vb, Kn, Cb, (Fn, I_b)
 eR, c, e, Kn, Rc, (En, V_c) , (Fn, I_c)
 --, d, f
 eL, e, f, Fn, Id, Kn, Ld, (En, V_d)
 eR, e, f, Sh, 0.6, Kn, RL, (En, V_L) , (Fn, I_L)

The names of the variables used in the POG Modeling Program can also have subscripts. See, for example, the “ V_a ” and “ V_c ” variables in the above POG system. The subscripts can be used in the plot only if the general parameter “Us, Si” is used.

The two general parameters “FnK, 14”, “CIK, b”, when reported in a system command-line, can be used to change the font size (FnK) and the color (CIK) of the name of the parameter “ \mathcal{K} ” for all the Physical Elements in the system:



**, As, No, Sn, No, Us, Si, FnK, 14, CIK, b

eV, a, b, An, -90, En, V_in, Fn, I_in

eR, a, c, Kn, R_a, En, V_a, Fn, I_a

--, b, d

eC, c, d, En, V_b, Kn, C_b, Fn, I_b, FnK, 20, CIK, r

eR, c, e, Kn, R_c, En, V_c, Fn, I_c

--, d, f

eL, e, f, Kn, L_d, Fn, I_d, En, V_d

eR, e, f, Sh, 0.6, Kn, R_L, En, V_L, Fn, I_L

When the same parameters “FnK, 20”, “CIK, r” are applied to a particular Physical Element (the “eC” element in this case), the general parameters “FnK, 14, CIK, b” are overwritten, and the new values are used.

The following “general” / “element” parameters can be used:

- parameters “FnK, nr” and “CIK, x” change the font size and the color of “ \mathcal{K} parameters”.
- parameters “FnX, nr” and “CIK, x” change the font size and the color of “input power variables”.
- parameters “FnY, nr” and “CIY, x” change the font size and the color of “output power variables”.

The general parameter “**As, Si**”, tells the POG Modeling Program to analyze the Physical System and **provide the differential equations of the Physical System in a symbolic form**. With reference to the considered electrical system one obtains the following output:

State space equations:

$$L \cdot \dot{X} = A \cdot X + B \cdot U$$

$$Y = C \cdot X + D \cdot U$$

Energy matrix L:

$$\begin{bmatrix} C_b, & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0, & L_d \end{bmatrix}$$

Power matrix A:

$$\begin{bmatrix} -1/(R_L + R_c) - 1/R_a, & -R_L/(R_L + R_c) \end{bmatrix}$$

$$\begin{bmatrix} R_L/(R_L + R_c), & -(R_L \cdot R_c)/(R_L + R_c) \end{bmatrix}$$

Input matrix B:

$$\begin{bmatrix} 1/R_a \end{bmatrix}$$

$$\begin{bmatrix} 0 \end{bmatrix}$$

Output matrix C:

$$\begin{bmatrix} -1/R_a, & 0 \end{bmatrix}$$

Input-output matrix D:

$$\begin{bmatrix} 1/R_a \end{bmatrix}$$

State vector X:

$$\begin{bmatrix} V_b \end{bmatrix}$$

$$\begin{bmatrix} I_d \end{bmatrix}$$

Input vector U:

$$\begin{bmatrix} V_{i_n} \end{bmatrix}$$

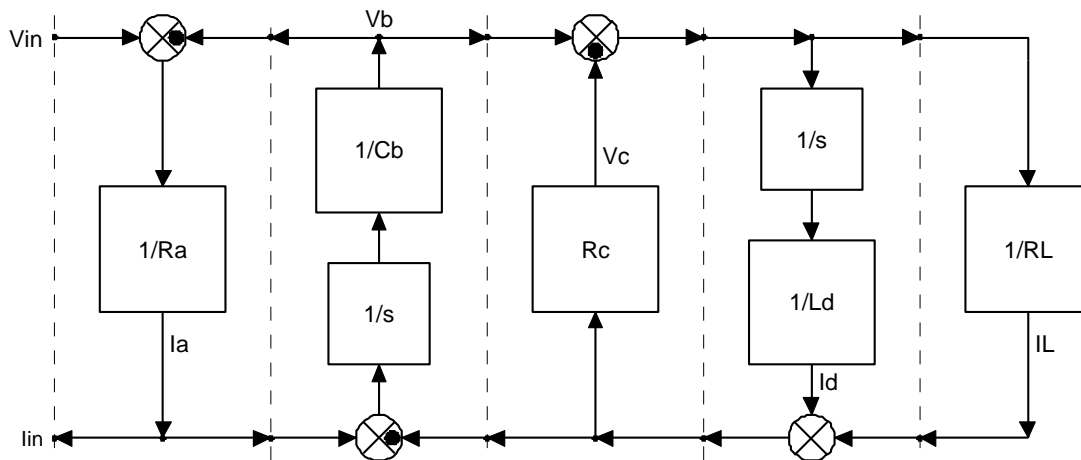
Output vector Y:

$$\begin{bmatrix} I_{i_n} \end{bmatrix}$$

The general parameter “**EQN, Si**”, tells the POG Modeling Program to **write the differential equations** in a file named “*_EQN.txt” where “*” stands for the name of the considered system.

Note: matrices **L**, **A**, **B**, **C** and **D** are function of all the parameters “**K**” of the physical elements only, and the state vector **X** contains the output power variables of all the dynamic elements of the considered System.

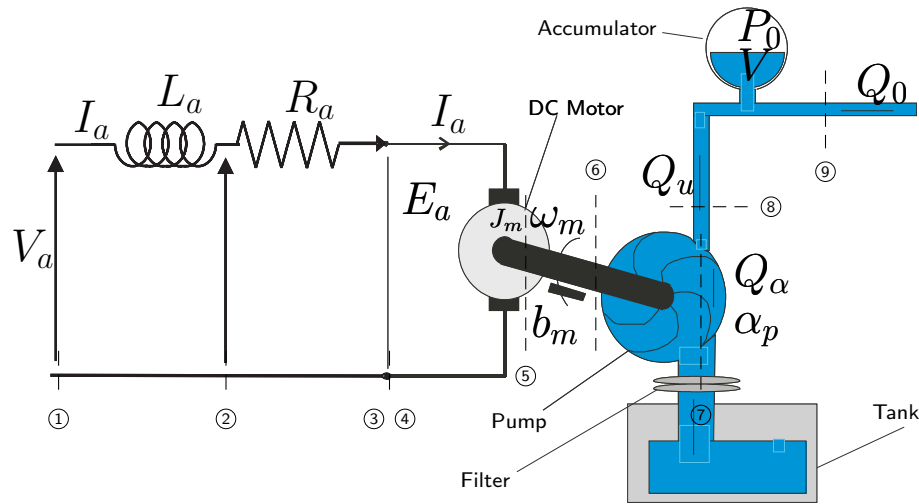
The general parameter “**POG, Si**”, tells the POG Modeling Program to **compute**, if it is possible, the **dynamic POG block scheme** of the considered Physical System:



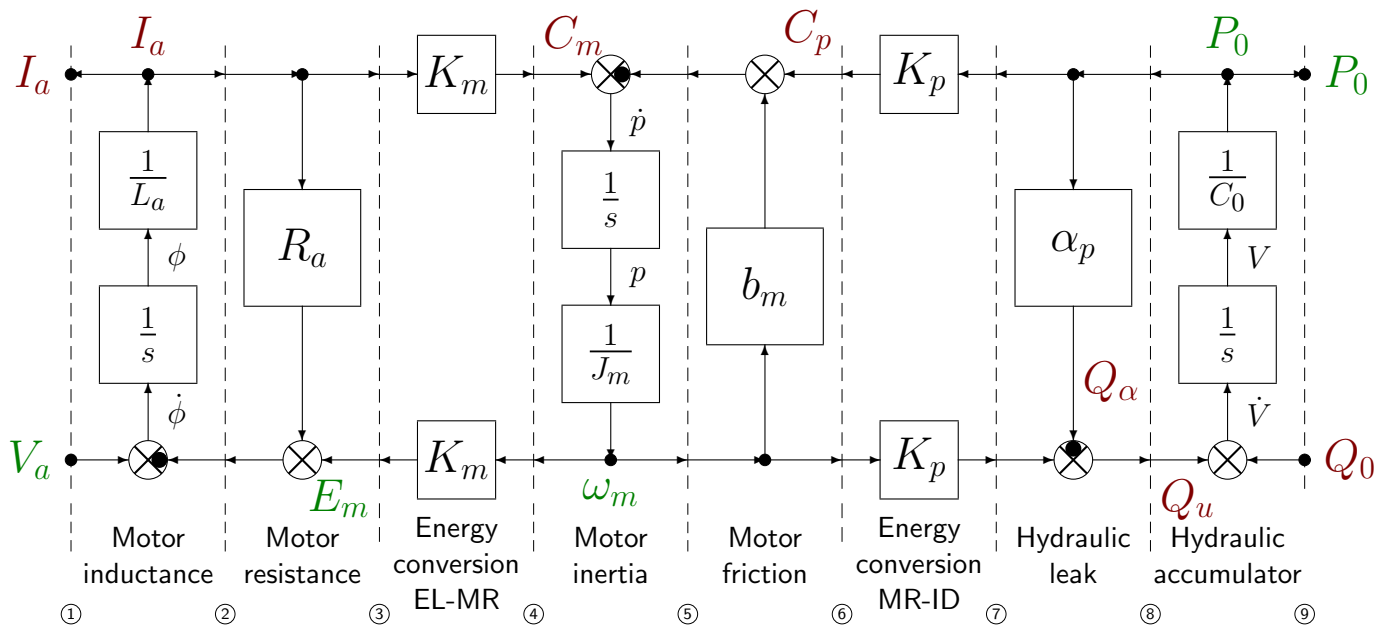
The general parameter “**pPr, Si**”, tells the POG Modeling Program to **print the image of the POG block scheme** in a file named “*_POG.eps”. By default, the output format of the image is “eps” (encapsulated postscript).

The output format of the image can be changed to “**png**”, “tiff” or “jpg”, using the general parameters “**pGTy, png**”, “pGTy, tiff” or “pGTy, jpeg”, respectively.

Example. A DC motor connected with an hydraulic pump:



The corresponding POG block scheme is:



The corresponding state space differential equations are:

$$\underbrace{\begin{bmatrix} L_a & 0 & 0 \\ 0 & J_m & 0 \\ 0 & 0 & C_0 \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} \dot{I}_a \\ \dot{\omega}_m \\ \dot{P}_0 \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} -R_a & -K_m & 0 \\ K_m & -b_m & -K_p \\ 0 & K_p & -\alpha_p \end{bmatrix}}_{-\mathbf{A}} \underbrace{\begin{bmatrix} I_a \\ \omega_m \\ P_0 \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} V_a \\ Q_0 \end{bmatrix}}_{\mathbf{u}}$$

Both the POG block scheme and the differential equations can be obtained using the POG Modeling Program as described in the following slides.

The electric part of the DC motor is modeled as follows:

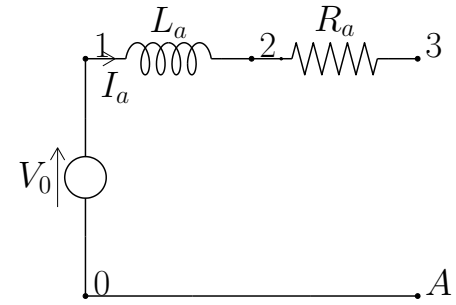
******, Sn, Si, Lw, 0.7

eV, 1, 0, An, -90, En, V_0, Fn, I_0

eL, 1, 2, Ln, 0.7, Zm, 0.9, Kn, L_a, Fn, I_a

eR, 2, 3, Ln, 0.7, Zm, 0.9, En, V_r_a, Kn, R_a

--, 0, A, Ln, 1.4



The commands have the following meaning:

******, Sn, Si, Lw, 0.7

******, Sn, Si shows the names of the nodes in the POG physical scheme;

******, Lw, 0.7 Sets the width of the lines in the POG physical scheme: "Lw=0.7";

eV, 1, 0, An, -90, En, V_0, Fn, I_0

eV, 1, 0, An, -90 inserts a voltage generator "eV" between nodes "1"=(0,1) and "0"="0"+Ln e^{An} 1i="0"+1e⁻⁹⁰ 1i="0" +(0,-1)=(0,0);

En, V_0 sets the name of the across variable: "En=V_0";

Fn, I_0 sets the name of the through variable: "Fn=I_0";

eL, 1, 2, Ln, 0.7, Zm, 0.9, Kn, L_a, Fn, I_a

eL, 1, 2 inserts an inductor "eL" between nodes "1"=(0,1) and "2"="1" + Ln e^{An} 1i="1"+0.7e⁰ 1i=(0,1)+(0.7,0)=(0.7,1);

Ln, 0.7 sets the length of the physical element: "Ln=0.7";

Zm, 0.9 sets the zoom of the physical element: "Zm=90%";

Kn, L_a sets the name of the internal parameter: "Kn=L_a";

Fn, I_a sets the name of the through variable: "Fn=I_a";

eR, 2, 3, Ln, 0.7, Zm, 0.9, En, V_r_a, Kn, R_a

eR, 2, 3 inserts a resistor "eR" between nodes "2"=(0.7,1) and "3"="2" + Ln e^{An} 1i="2"+0.7e⁰ 1i=(0.7,1)+(0.7,0)=(1.4,1);

Ln, 0.7 sets the length of the physical element: "Ln=0.7";

Zm, 0.9 sets the zoom of the physical element: "Zm=90%";

En, V_r_a sets the name of the across variable: "En=V_r_a";

Kn, R_a sets the name of the internal parameter: "Kn=R_a";

--, 0, A, Ln, 1.4

--, 0, A inserts a connection line "--" between nodes "0"=(0,0) and "A"="0" + Ln e^{An} 1i="0"+1.4e⁰ 1i=(0,0)+(1.7,0)=(1.4,0);

Ln, 1.4 sets the length of the physical element: "Ln=1.4";

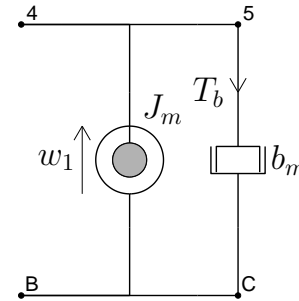
The following **CB**-command-line will be explained later in this section:

```
'CB, [3; 4], [A; B], Kn, F2=K_m*F1, En, [E_m;w_2], ...'
```

This block defines the positions of point “4”= $(2.5,1)$ and point “B”= $(2.5,0)$.

The mechanical part of the DC motor is modeled by using the following commands:

```
rJ, 4, B, Sh, 0.4, Kn, J_m, En, w_1
--, 4, 5, Ln, 0.8
--, B, C, Ln, 0.8
rB, 5, C, Kn, b_m, Fn, T_b
```



The commands have the following meaning:

rJ, 4, B, Sh, 0.4, Kn, J_m, En, w_1

rJ, 4, B inserts a rot. mechanical element "rJ" between the nodes "4" and "B":

Sh, 0.4 sets the value of the lateral shift: "Sn=0.4";

Kn, J_m sets the name of the internal parameter: "Kn=J_m":

En, w_1 sets the name of the across variable: "En=w_1";

--, 4, 5, Ln, 0.8

--, 4, 5 inserts a connection line "--" between the nodes "4"=(2.5,1) and "5"=
"4"+0.8e⁰¹ⁱ=(2.5,1)+(0.8,0)=(3.3,1);

Ln, 0.8 sets the length of the physical element: "Ln=0.8"

--, B, C, Ln, 0.8

--, B, C inserts a connection line "--" between the nodes "B"=(2.5,0) and "C"=
"B"+0.8e⁰¹ⁱ=(2.5,0)+(0.8,0)=(3.3,0);

Ln, 0.8 sets the length of the physical element: "Ln=0.8"

rB, 5, C, Kn, b_m, Fn, T_b

rB, 5, C inserts a friction element “rB” between the nodes “5” and “C”;

Kn, b_m sets the name of the internal parameter: "Kn=b_m"

F_n, T_b sets the name of the through variable: “F_n=T_b”

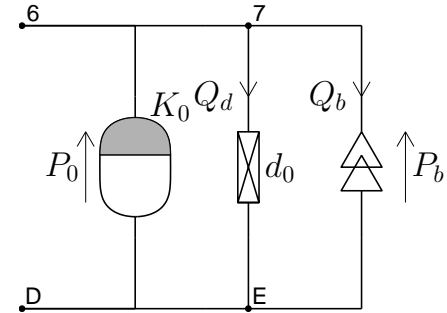
The following CB-command-line will be explained later on in this section:

CB, [5; 6], [C; D], Kn, F1=K_p*E2, En, [w_1;p_2], ...
... Fn, [T_p;Q_p], Sh, [0.5;-0.2], La,0.5

This block defines the positions of point “6”=(4.4,1) and point “D”=(4.4,0).

The hydraulic part of the DC motor is modeled by using the following commands:

```
iK, 6, D, Sh, 0.4, Kn, K_0, En, P_0
--, 6, 7, Ln, 0.8
--, D, E, Ln, 0.8
iG, 7, E, Kn, d_0, Fn, Q_d
iQ, 7, E, En,-P_b,Fn,Q_b,Sh,0.4,Pin,1,ShY,Si
```



The commands have the following meaning:

iK, 6, D, Sh, 0.4, Kn, K_0, En, P_0

iK, 6, D inserts an hydraulic stiffness “iK” between the nodes “6” and “D”;

Sh, 0.4 sets the value of the lateral shift: “Sn=0.4”;

Kn, K_0 sets the name of the internal parameter: “Kn=K_0”;

En, P_0 sets the name of the across variable: “En=P_0”.

--, 6, 7, Ln, 0.8

--, 6, 7 inserts a connection line “--” between the nodes “6” and “7”=(5.2,1);

Ln, 0.8 sets the length of the physical element: “Ln=0.8”

--, D, E, Ln, 0.8

--, D, E inserts a connection line “--” between the nodes “D” and “E”=(5.2,0);

Ln, 0.8 sets the length of the physical element: “Ln=0.8”

iG, 7, E, Kn, d_0, Fn, Q_d

iG, 7, E inserts a hydraulic conductance “iG” between the nodes “7” and “E”;

Kn, d_0 sets the name of the internal parameter: “Kn=d_0”

Fn, Q_d sets the name of the through variable: “Fn=Q_d”

iQ, 7, E, Sh, 0.4, En,-P_b, Fn, Q_b, Pin, 1, ShY, Si

iQ, 7, E inserts a volume flow rate generator “iQ” between the nodes “7” and “E”;

Sh, 0.4 sets the value of the lateral shift: “Sn=0.4”;

En,-P_b sets the name of the across variable: “En=P_b”. The minus sign means that the variable must be plotted on the opposite side of the element;

Fn, Q_b sets the name of the through variable: “Fn=Q_b”;

Pin, 1 sets the name of the positive power direction: “Pin=1”. If “Pin=1” the positive power enters the element, while if “Pin=-1” it exits from the element;

ShY, Si sets the value of the “show the output variable”: “ShY=Si”. In this case, only for the “iQ” element the output variable Q_b is shown.

Let us now consider the following command-line :

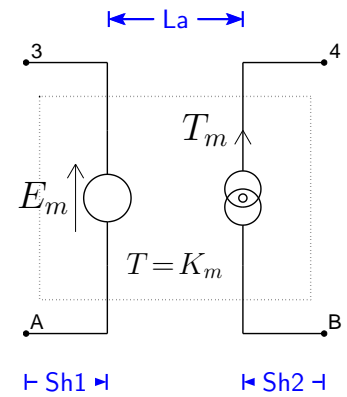
**CB, [3; 4], [A; B], Kn, F2=K_m*F1, En, [E_m;w₂], ...
... Fn, [I₁;T_m], Sh, [0.3;-0.3], La, 0.5**

The “**CB**” block, i.e. the “Connection Block”, is a special POG block that allows to connect two different Energetic Domains. This block converts the power flows within the system from one type of power to another, without storing or dissipating energy. This is a crucial element for modeling physical systems that involves two or more Energetic Domains. The electric DC model that we are modeling, for example, is a physical system that converts electric power into mechanical rotational power, and viceversa.

A “Connection Block” “**CB**” is a “double” element which is defined between two couples of points:

CB, [3; 4], [A; B]

In this case: a) the [3; 4] are the “top” points and [A; B] are the “bottom” points; b) the position of points “3” and “A” has already been defined, while the position of points “4” and “B” has not yet been defined.



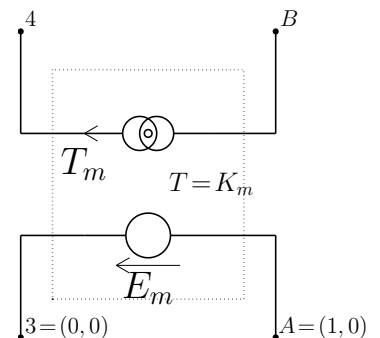
The “**CB**” block is composed of two branches. In this case the first branch is connected to points “3” and “A” and the second branch is connected to points “4” and “B”. The position of the new points “4” and “B” is computed as follows:

1. the line “4-B” is parallel to line “3-A”.
2. the distance “Dis” between the two lines “3-A” and “4-B” is computed as follows: $Dis = Sh1 + La - Sh2$ where “Sh1” is the lateral shift of the first branch of the CB block, “La” is the lateral distance between the two branches and “Sh2” is the lateral shift of the second branch of the CB block.
3. the default values are: “Sh1=0.4”, “La=0.4” and “Sh2=-0.4”.
4. the values of these parameters can be changed using commands “Sh, [Sh1; Sh2]” and “La, d”. In the above example the following values have been used: **Sh, [0.2; -0.2], La, 0.3**.

If the position of points “3” and “A” is defined as follows: “3”=(0,0) e “A”=(1,0), the command-line:

CB, [3; 4], [A; B]

generates a “vertical” CB block (see the aside figure). In this case, “Dis”=0.4+0.4-(-0.4)=1.2 (default values) and therefore “4”=(0,1.2) and “B”=(1,1.2).



The parameter “**Kn, F2=K_m*F1**” is a crucial element for the CB block because it defines which mathematical relations exist between the four power variables involved in the CB. The names of these power variables can be defined using the “En” and “Fn” commands.

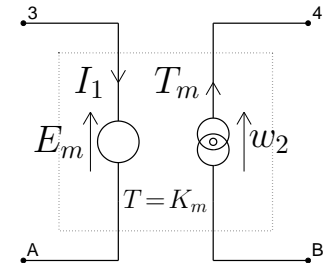
The ‘**En, [E_m; w_2]**’ command defines the name of the two **across-power variables**: the **across-variable** “**E_m**” is the electromotive voltage in the electric part of the motor (branch “3-A”) and the **across-variable** “**w_2**” is the angular velocity of the mechanical part of the electric motor (branch “4-B”).

The ‘**Fn, [I_1; T_m]**’ command defines the name of the two **through-power variables**: the **through-variable** “**I_1**” is the armature current of the electric part of the motor (branch “3-A”) and the **through-variable** “**T_m**” is the motor torque acting on the mechanical part of the electric motor (branch “4-B”).

All the four power-variables can be shown in the POG scheme using the general parameters “ShX” and “ShY”:

****, ShX, Si, ShY, Si**

The obtained graphical result is shown in the aside figure.



The string “**F2=K_m*F1**” present in the command “**Kn, F2=K_m*F1**” defines the CB as a “Transformer” and defines the name of the internal parameter “**K**” as “**K_m**”.

A CB block is a “**Transformer**” if it transforms the **across-variable/Effort** and the **through-variable/Flow** into power-variables of the same type. The string “**T=K_m**” plotted in the center of the CB block (see the above figure) denotes the CB block as a **Transformer** characterized by the internal parameter “**K_m**”.

The string “**F2=K_m*F1**” means that the Flow variable **F2** of the second branch of the CB block is obtained multiplying the Flow variable **F1** of the first branch by the internal parameter “**K_m**”. In the considered case:

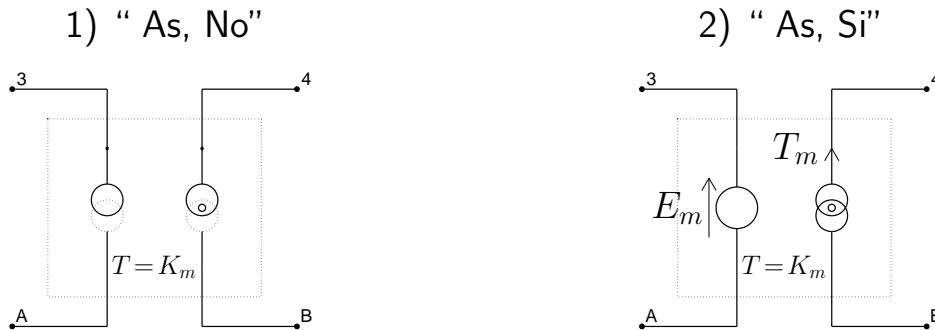
$$\mathbf{F2} = \mathbf{K_m} * \mathbf{F1} \quad \Rightarrow \quad T_m = K_m I_1, \quad E_m = K_m w_2.$$

The relation **F2=K_m*F1** also implies the “dual” relation “**E1=K_m*E2**” in a biunivocal way. The string “**E1=K_m*E2**” means that the Effort variable **E1** of the first branch of the CB is obtained multiplying the Effort variable **E2** of the second branch by the internal parameter “**K_m**”.

So, the following two command-lines produce the same final result:

CB, [3; 4], [A; B], Kn, F2=K_m*F1 and **CB, [3; 4], [A; B], Kn, E1=K_m*E2** .

The graphical representation of the **CB** block changes when the system is analyzed. When “As, No” and “As, Si”, respectively, the following two graphical representations are obtained:



In the first case, **As, No**, the POG Modeling Program detects the Energetic Domains connected to the two inputs and plots the corresponding “Undefined Generators”. For the considered Energetic Domains, the “Undefined Generators” are the following:

	Electric	Mecc. Tras.	Mecc. Rot.	Hydraulic
Undefined-Generators \mathcal{G}_f	<p style="text-align: center; color: magenta;">eU</p> <p style="text-align: center;">Current Un.Gen.</p>	<p style="text-align: center; color: magenta;">mU</p> <p style="text-align: center;">Force Un.Gen.</p>	<p style="text-align: center; color: magenta;">rU</p> <p style="text-align: center;">Torque Un.Gen.</p>	<p style="text-align: center; color: magenta;">iU</p> <p style="text-align: center;">Flow Rate Un.Gen.</p>

In the second case, **As, Si**, the system is analyzed and the graphical representation of the **CB** block given by the POG Modeling Program depends on its dynamic orientation.

Given the following commands:

******, As, Si, POG, Si

el, 1, A, An,-90, Sh, -0.3, En, E_1, Fn, F_1 (Electrical domain)

CB, [1; 2], [A; B], Kn, **F2=K_m*F1**, En, [E_1;-E_2], Fn, [F_1;F_2]

rW, 2, B, Sh, 0.3, En, -E_2, Fn, F_2, Pin, 1 (Rot. Mechanical domain)

one obtains the following output (**Transformer**):

Physical scheme	POG scheme	State space equations:
		$\begin{bmatrix} E_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 0 & K_m \\ K_m & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ E_2 \end{bmatrix}$

Given the following commands:

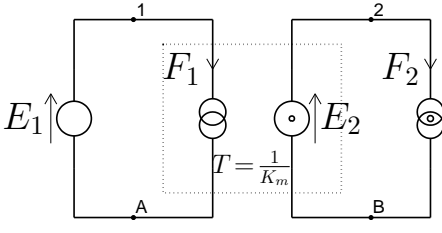
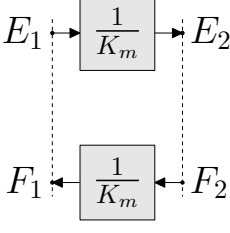
******, As, Si, POG, Si

eV, 1, A, An,-90, Sh, -0.3, En, E_1, Fn, F_1 (Electrical domain)

CB, [1; 2], [A; B], Kn, F2=K_m*F1, En, [E_1;-E_2], Fn, [F_1;F_2]

rT, 2, B, Sh, 0.3, En, -E_2, Fn, F_2, Pin, 1 (Rot. Mechanical domain)

one obtains the following output (**Transformer**):

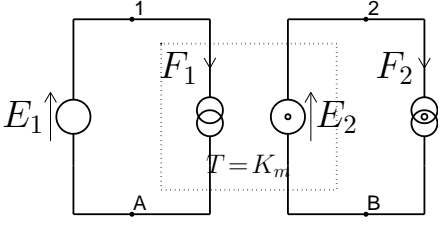
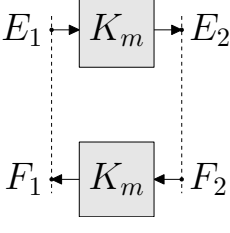
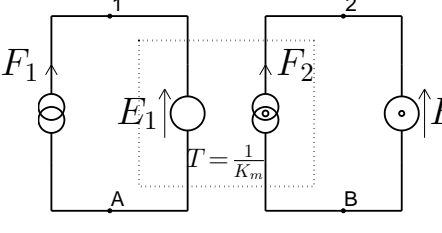
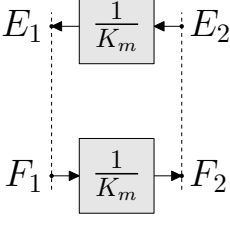
Physical scheme	POG scheme	State space equations:
		$\begin{bmatrix} F_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{K_m} \\ \frac{1}{K_m} & 0 \end{bmatrix} \begin{bmatrix} E_1 \\ F_2 \end{bmatrix}$

In the previous two examples, the same results can also be obtained substituting the string **F2=K_m*F1** with the string **E1=K_m*E2**.

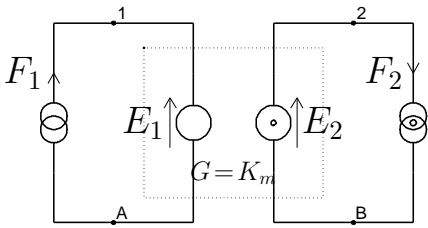
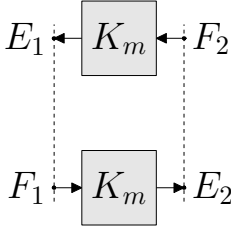
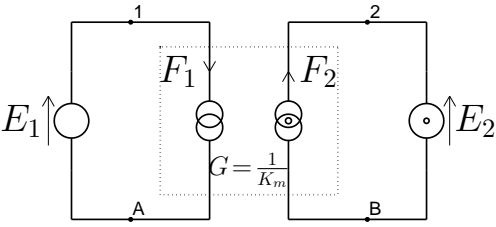
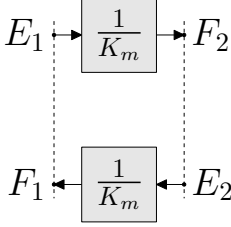
Note: the flow variables **F1**, **F2** and the effort variables **E1**, **E2** are still linked by relations **F2=K_m*F1** and **E1=K_m*E2**, respectively.

Note: in the last two examples, the string **F2=K_m*F1** in the **CB**-command-line is the same but, due to different inputs, the **CB** block has been oriented in a different way, and therefore the final graphical representations are different.

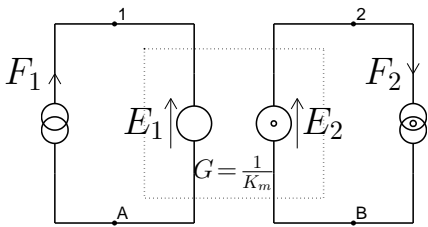
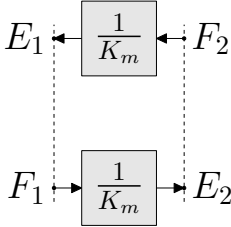
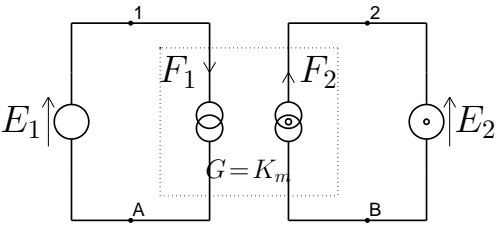
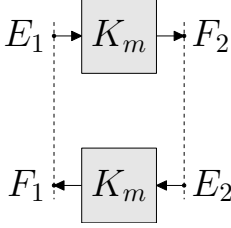
Using the string **F1=K_m*F2** or the string **E2=K_m*E1** in the **CB** block one obtains one of the following two outputs (**Transformer**):

Physical scheme	POG scheme	State space equations:
		$\begin{bmatrix} F_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} 0 & K_m \\ K_m & 0 \end{bmatrix} \begin{bmatrix} E_1 \\ F_2 \end{bmatrix}$
		$\begin{bmatrix} E_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{K_m} \\ \frac{1}{K_m} & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ E_2 \end{bmatrix}$

Using the string **E2=K_m*F1** or the string **E1=K_m*F2** in the **CB** block, one obtains one of the following two outputs (**Gyrator**):

Physical scheme	POG scheme	State space equations:
		$\begin{bmatrix} E_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} 0 & K_m \\ K_m & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$
		$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{K_m} \\ \frac{1}{K_m} & 0 \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \end{bmatrix}$

Using the string **F1=K_m*E2** or the string **F2=K_m*E1** in the **CB** block, one obtains one of the following two outputs (**Gyrator**):

Physical scheme	POG scheme	State space equations:
		$\begin{bmatrix} E_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{K_m} \\ \frac{1}{K_m} & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$
		$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 0 & K_m \\ K_m & 0 \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \end{bmatrix}$