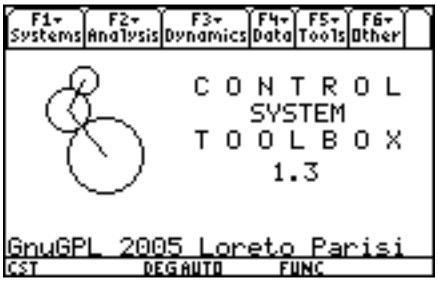
Control System Toolbox

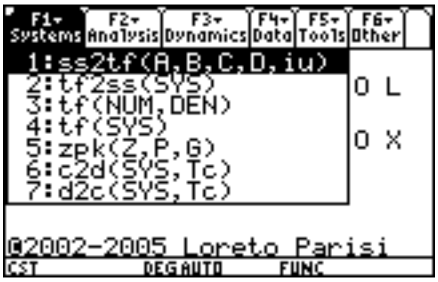
for TI-89

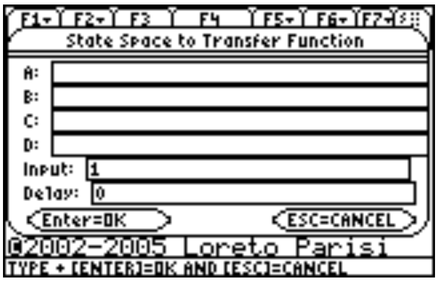


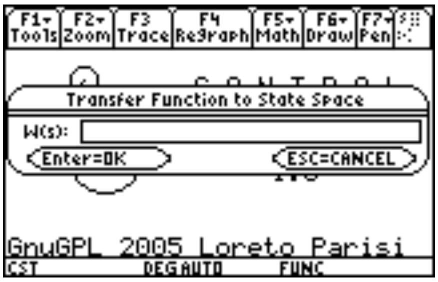
release 1.3

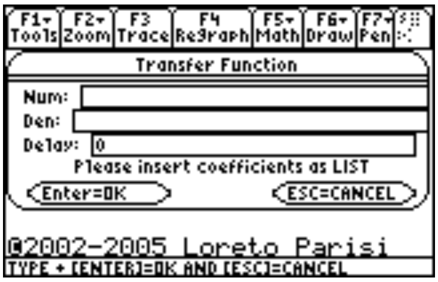
The CST Reference Guide

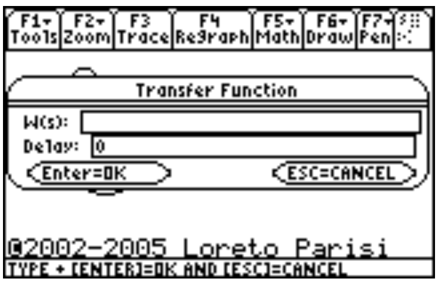
First Edtion October 2005

<table>
<tr><th colspan="2">Systems</th></tr>
<tr><td>


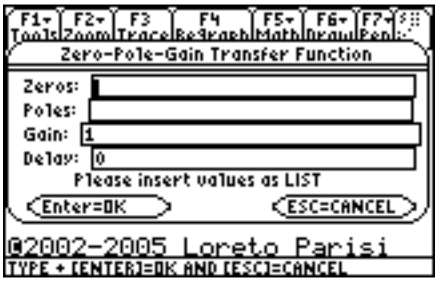
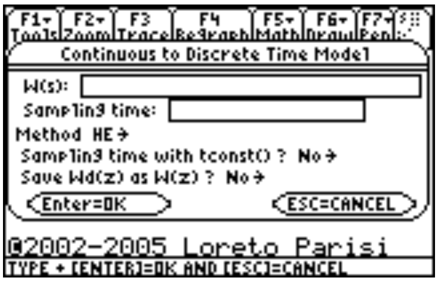
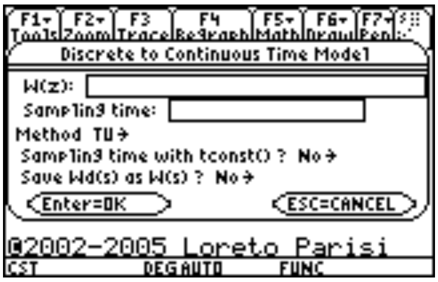
</td><td>The Systems menu (F1) contains all the functions to build the model, using state-space or transfer function and to perform conversions from one representation to another, even from continuous time to discrete time model.</td></tr>
<tr><td>


</td><td>ss2tf(A,B,C,D,iu)
Gives transfer function W(s)=$C(sI-A)^{-1}B+D$ from state-space $\dot{x} = Ax + Bu$, $y = Cx + Du$, relating to input iu (it works on MIMO systems, but only one input at time).
Delay τ is the Time Delay $e^{-\tau s}$.</td></tr>
<tr><td>


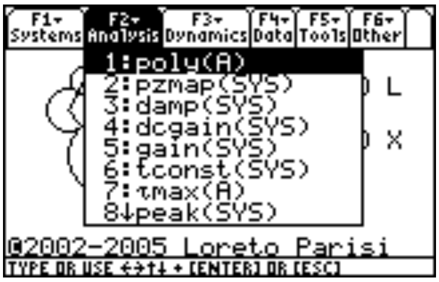
</td><td>tf2ss(SYS)
Convert transfer function SYS in the state-space representation $\dot{x} = Ax + Bu$, $y = Cx + Du$, using the observability canonical form.
Delay τ is the Time Delay $e^{-\tau s}$.</td></tr>
<tr><td>


</td><td>tf(NUM,DEN)
Calculates transfer function, where NUM and DEN are LIST of coefficients of numerator’s and denominator’s polynomial: NUM={b_0,b_1,…,b_n}, DEN={a_0,a_1,…a_n}, so W(s)=$\dfrac{b_0 s^n + b_1 s^{n-1} + \ldots + b_n}{a_0 s^n + a_1 s^{n-1} + \ldots + a_n}$.
Delay τ is the Time Delay $e^{-\tau s}$.</td></tr>
<tr><td>


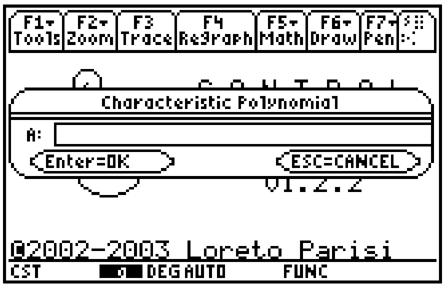
</td><td>tf(SYS)
Calculates transfer function from a rational expression in s
Delay τ is the Time Delay $e^{-\tau s}$.</td></tr>
</table>

	***zpk(Z,P,G)*** Calculates transfer function W(s) in the zeros-poles-gain representation, where Z, P are LIST of zeros of numerator and denominator (poles), while G is NUM and represents constant gain K. Control's Toolbox v.1.16 author: Francesco Orabona E-mail: bremen79@infinito.it Home:http://web.genie.it/utenti/b/bremen79/
	***c2d(SYS,T_c)*** Converts continuous time model SYS to the discrete time model, using sample time T_c and different methods: HE (Linear Hold Equivalence), TU (Bilinear Tustin), BE (Bilinear Backward Eulero), FE (Bilinear Forward Eulero). Can use function tconst(SYS) to determinate sample time T_c. Use function sampler(A,B,T_c) to use ZOH method. Can save the resulting discrete time model Wd(z) as current discrete transfer function W(z).
	***d2c(SYS,T_c)*** Converts discrete time model SYS (in z) to the continuous time model, using sample time T_c and different methods: HE (Linear Hold Equivalence), TU (Bilinear Tustin), BE (Bilinear Backward Eulero), FE (Bilinear Forward Eulero).Can use function tconst(SYS) to determinate sample time T_c. Can save the resulting continuous time model Wd(s) as current continuous transfer function W(s).

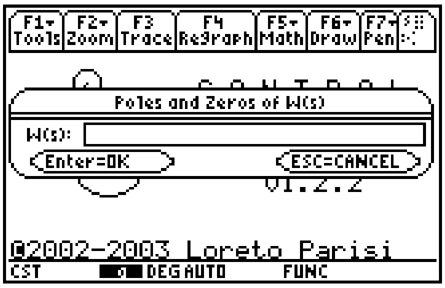
Analysis



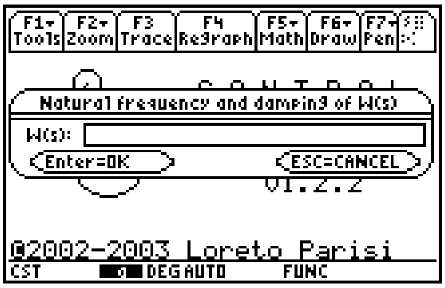
The ***Analysis*** menu (F2) contains all the tools to analyze the model you have created with Systems' tools. You can also analyze different models, using different SYS at time.
This will not change current transfer function.



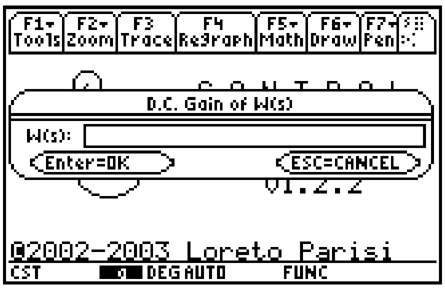
poly(A)
This function calculates characteristic polynomial of matrix A, as p(s)=|sI-A|, where |•| is determinat of a matrix.



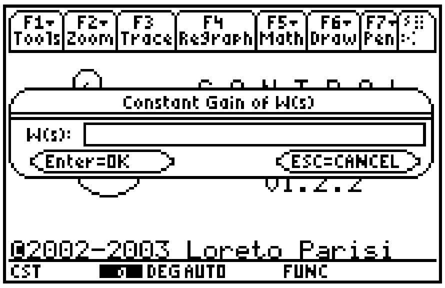
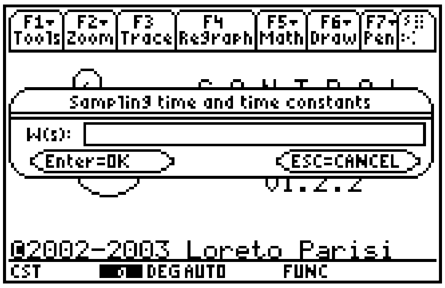
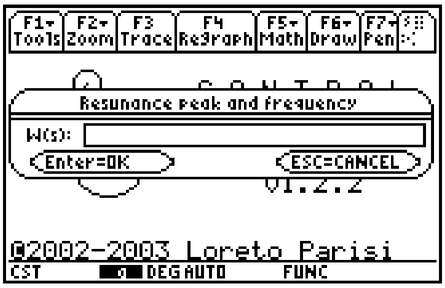
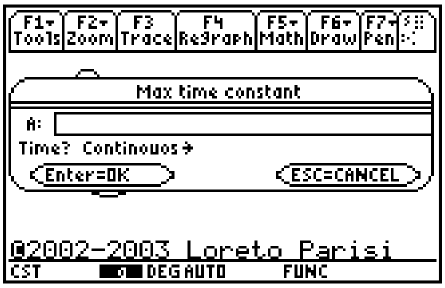
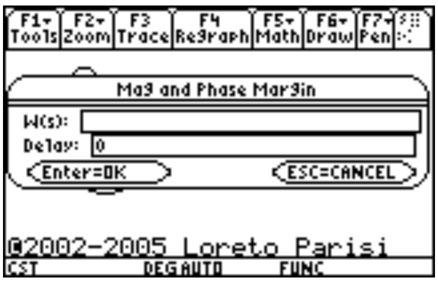
pzmap(SYS)
This function calculates poles and zeros of given transfer function SYS, where poles are zeros of denominator of SYS.

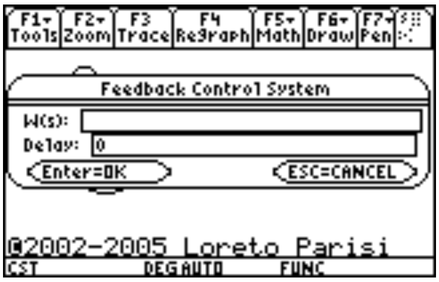


damp(SYS)
Calculate natural frequencies ω_{nh} and damping factors ζ_h for transfer function SYS, where $\omega_{nh}=\sqrt{\alpha_h^2+\omega_h^2}$ and $\zeta_h=\frac{-\alpha_h}{\omega_{nh}}$ for eigenvalue $\lambda_h=\alpha_h+j\omega_h$.



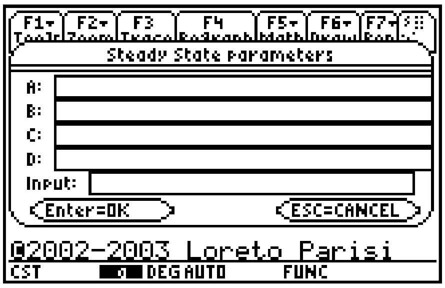
dcgain(SYS)
Calculates d.c. gain G for transfer function SYS, as $G=\lim_{s\to 0} W(s)$.

<table>
<tr>
<td>
</td>
<td>gain(SYS)
Calculates constant gain K for transfer function SYS, as K=$\lim_{s\to 0} s^{n_0 - m_0} W(s)$, where n_0 and m_0 are multiplicity of zero roots for denominator and numerator.</td>
</tr>
<tr>
<td>
</td>
<td>tconst(SYS)
Calculates sample time T_c and time constants τ_i, τ_h, and T_h, where $\tau_i = -\frac{1}{\lambda_i}$, $\tau_h = -\frac{1}{\alpha_h}$ and $T_h = \frac{2\pi}{\omega_h}$, while T_c= $0.1\min\{\tau_i , \tau_h , T_h\}$.</td>
</tr>
<tr>
<td>
</td>
<td>peak(SYS)
This function uses a proprietary numerical algorithm
to calculate resonance peak $M_p=\max_\omega M(\omega)$, where $M(\omega)=|W(s)|_{s=j\omega}$ and relating frequency f_r, which is $M(2\pi f_r)=M_P$.</td>
</tr>
<tr>
<td>
</td>
<td>tmmax(A)
Calculates maximum time constant for characteristic polynomial of matrix A, in continuous or discrete time, where $\tau_{max} = \frac{1}{\min(-\Re\lambda_i)}$ (continuous time) and $\tau_{max} = \frac{1}{\min(\ln|\lambda_i|)}$ (discrete time).</td>
</tr>
<tr>
<td>
</td>
<td>margin(SYS)
Calculates Mag and Phase Margins.
$K_m= 1/|W(i\omega_m)|$ (Mag Margin)
ω_m: $\angle W(i\omega_m)= -180°$
$\phi_m= 180°- |\phi_c|$ (Phase Margin)
ω_c: $|W(i\omega_c)|=1$
$\phi_c= \angle W(i\omega_c)$ (Critical Phase)
$\tau_c= \phi_m/\omega_c*\pi/180°$ (Critical Time Constant)</td>
</tr>
</table>

	***feedback(sys)*** Performs the analysis and design of the closed loop control system of process SYS. Delay τ is the Time Delay $e^{-\tau s}$. Please see *Feedback Control Systems* section.

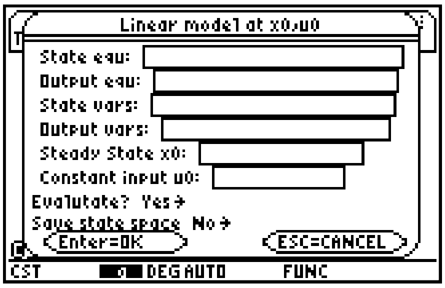
Dynamics

The ***Dynamics*** menu (F3) contains functions concerning dynamics of system for input, output and linearization of a non-linear model, frequency analysis with Bode and Nyquist diagrams and Root Locus yet.



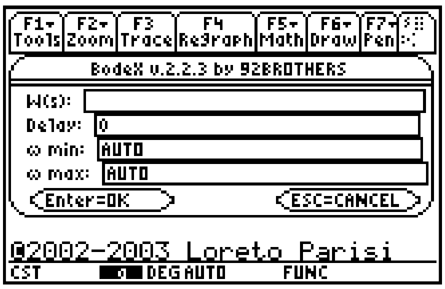
trim(A,B,C,D,u$_0$)

This function calculates the steady state x_0, relating to input u_0 for state-space $\dot{x} = Ax + Bu$, $y = Cx + Du$.



linmod(f,y,x,u,x$_0$,u$_0$)

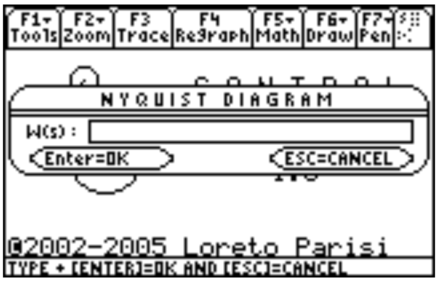
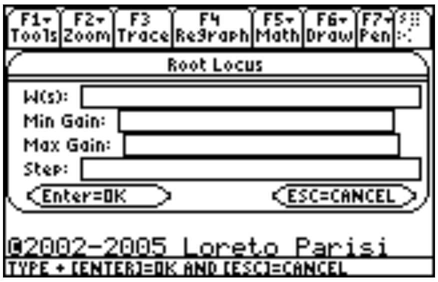
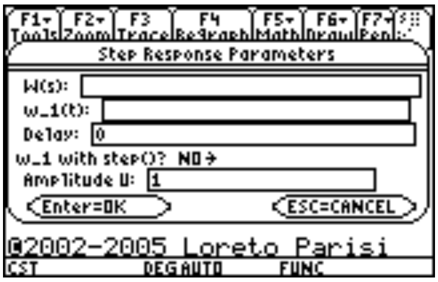
This function calculates linear model for non–linear model assigned in terms of input equations f, such as $f=\{f_1(x,u),\ldots,f_n(x,u)\}$ and output equations y, such as $y=\{y_1(x,u),\ldots y_n(x,u)\}$, relating to constant input u_0 and steady state x_0. The jacobian matrixes can be evalutated in x_0, u_0 and the state-space can be saved, or can be calculated in a symbolic way, before being evalutated.



bodex(SYS)

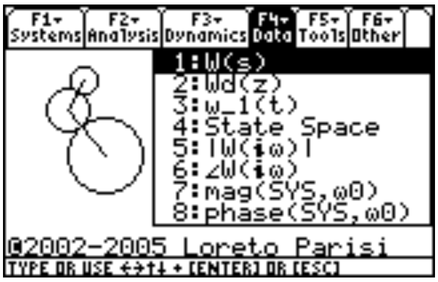
This program, made by 92BROTHERS, plots Bode diagrams of phase and magnitude and offers several tools to work with the plottoed diagrams.

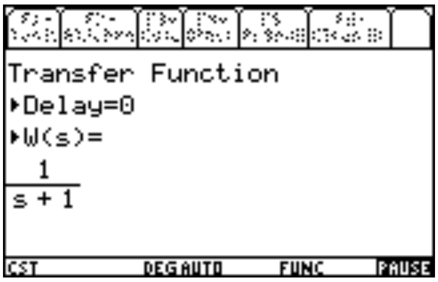
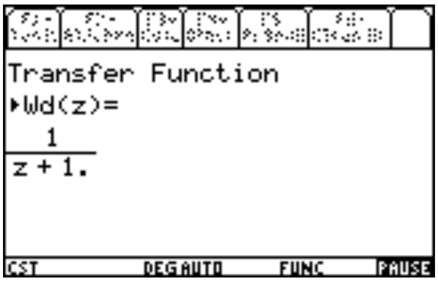
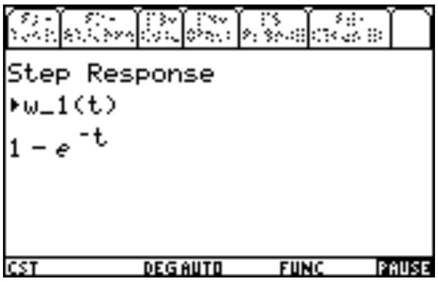
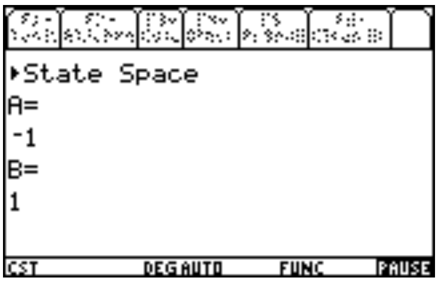
BodeX v.2.2.3
Copyright © 2000 92BROTHERS
Email: 92brothers@infinito.it
Home: http://www.92brothers.net/

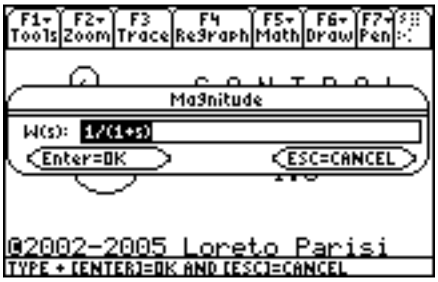
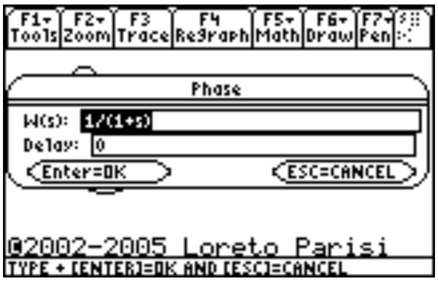
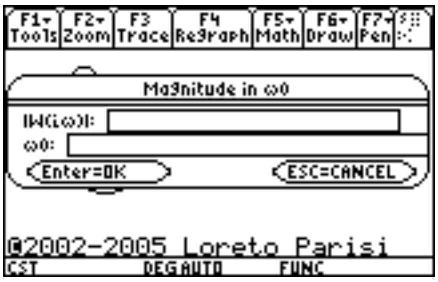
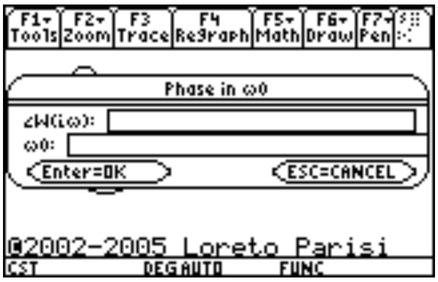
	***nyquist(SYS)*** Plots Nyquist diagram of SYS Control Toolbox v1.16 Author: Francesco Orabona E-mail: bremen79@infinito.it Home:http://web.genie.it/utenti/b/bremen79/
	***rlocus(SYS)*** Plots the Root Locus of SYS Max, min gain are the extremes of the gain list. Porting for CST: Loreto Parisi Control Toolbox v1.16 Author: Francesco Orabona E-mail: bremen79@infinito.it Home:http://web.genie.it/utenti/b/bremen79/
	***step(SYS)*** This tool calculates the step response for SYS, as $U^*w_{-1}(t) = L^{-1}(W(s)U/s)$, with amplitude U. Needs the tool LZT to perform symbolic calculation of Laplace direct and inverse transformation. LZT r7 Author: Jiri Bazant Email: georger@razdva.cz Home: http://www.razdva.cz/georger/
	***pstep(SYS)*** Calculates characteristic parameter of step response for transfer function SYS, such as T_e, T_a, T_s, T_p and s. Step response $w_{-1}(t)$ can be specified or calculated with step(SYS). Needs the tool LZT to perform symbolic calculation of Laplace direct and inverse transformation.

	***gstep(w_1(t))*** This tool plots the step response w_1(t) calculated with step(SYS) or specified directly. Can use pstep(SYS) to evalutate w_1(t) around its typical parameters.

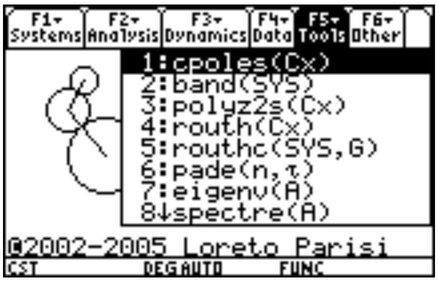
Notes.
1. About *pstep(SYS)*. Calculates time domain parameters of the step response for transfer function SYS, such as T_e, T_a, T_s, T_p and s, where T_e is the Elongation Time, T_r is the Raise Time, T_s is the Delay Time and s is the elongation.

<table>
<tr><th colspan="2">Data</th></tr>
<tr><td>
</td>
<td>The Data menu (F4) gives access to current transfer function W(•)[1], its discrete model Wd(•), the step response w-1(•), the current state space and magnitude and phase of W(•).

1 According to current Time Domain Settings (see Other menu)</td></tr>
<tr><td>
</td>
<td>W(s) [W(z)]
Displays the current transfer function. SYS refers to it in all calculations of current session of CST, once you've calculated it with one of the tools of Systems menu.</td></tr>
<tr><td>
</td>
<td>Wd(z) [Wd(s)]
The discrete transfer function Wd(z) [Wd(s)], obtained by c2d(SYS),Tc) [d2c(SYS,Tc)] using the current transfer function W(s) [W(z)] or by sampler(A,B,Tc) using the current state space.</td></tr>
<tr><td>
</td>
<td>w_1(t) [w_1(k)]
Shows the current step response obtained with step(SYS).</td></tr>
<tr><td>
</td>
<td>State Space
It displays the current state space, as defined from one of tools of System menu.</td></tr>
</table>

	***/ W(ω) /*** Displays magnitude of transfer function W(•) in Laplace domain (for W(s)) and even in domain Z (for W(z)).
	***∠W(ω)*** Displays phase of transfer function W(•) in Laplace domain (for W(s)) and even in domain Z (for W(z)).
	***mag(SYS,*ω_0*)*** Calculates magnitude of SYS in Laplace domain (for W(s)) and even in domain Z (for W(z)), relating to ω_0.
	***phase(SYS,*ω_0*)*** Calculates phase of SYS in Laplace domain (for W(s)) and even in domain Z (for W(z)), relating to ω_0.

Tools



The ***Tools*** menu (F5) offers several useful functions to complete the analysis of the model you're working and to give more detailed information about it. Moreover presents different tools for discrete systems and finite state systems.

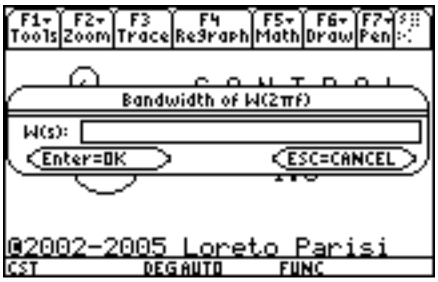
cpoles(Cx)

It calculates zeros of polinomyal given as LIST of coefficients, Cx.

polyz2s(Cx)

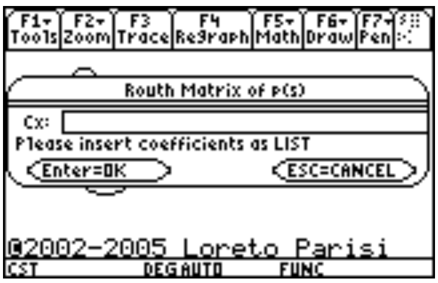
This tool calculates the continuous polynomial q(s), relating to discrete polynomial p(z), assigned in terms of its coefficients LIST, Cx, using the formula

q(s) = $(s-1)^n \, p(z)\Big|_{z=\frac{s+1}{s-1}}$



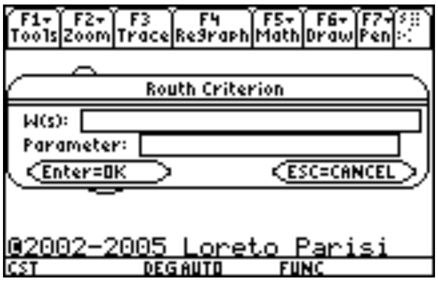
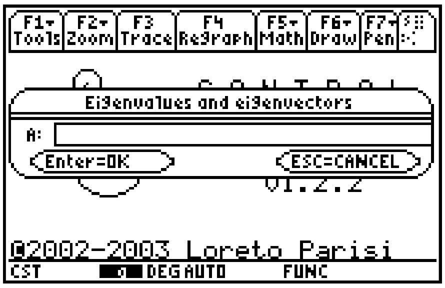
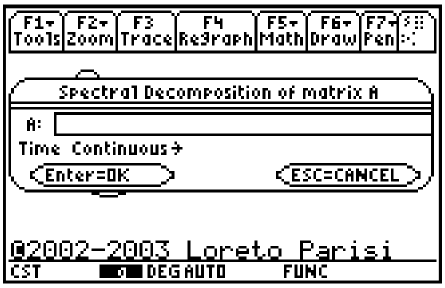
band(SYS)

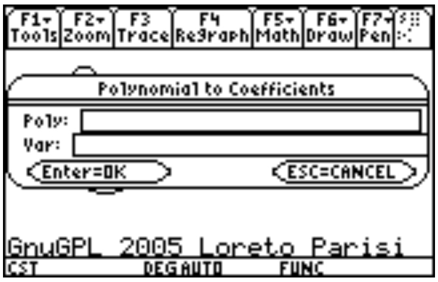
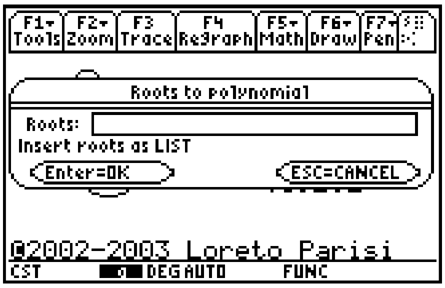
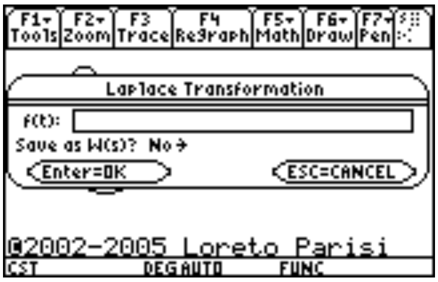
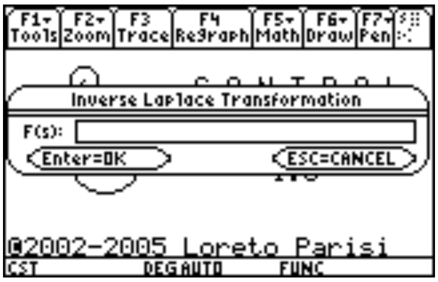
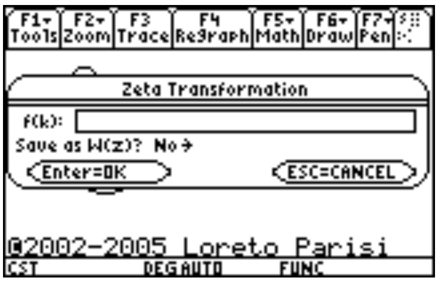
This function uses a numerical algorithm and several preexistent formulas to calculate bandwith of system with transfer function SYS. It calculates f_i, f_s, where B=[f_i,f_s], f_r (resonance frequency) and M_p (resonance peak).

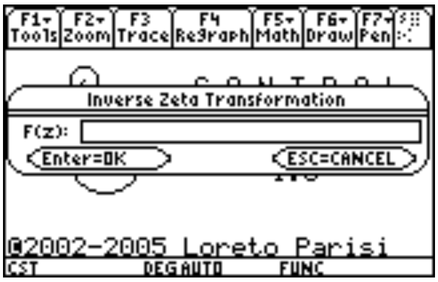


routh(Cx)

It calculates the Routh matrix for polynomial assigned with its coefficients LIST, Cx.

<table>
<tr><td></td><td>routhc(Cx)
Applies Routh Criterion to parametric W(s) to obtain Routh conditions using the specified parameter.</td></tr>
<tr><td></td><td>pade(n,τ)
Calculates the delay Padè approximation. Resulting delay transfer function can be saved as the current continuous transfer function W(s).</td></tr>
<tr><td></td><td>eingev(A)
It calculates eigenvalues and eigenvectors of matrix A.</td></tr>
<tr><td></td><td>spectre(A)
This tool calculates the spectral decomposition of matrix A, even in the continuous (e^{At}) and in the discrete time (A^k), relating to real eigenvalues and complex eigenvalues.[1]</td></tr>
<tr><td></td><td>sampler(A,B,T_c)
This function performs the discrete time conversion of continuous time model with state-space $\dot{x} = Ax + Bu$ at sample time T_c, using the ZOH (Zero Order Hold) method. It permits to use sample time T_c calculated with function tconst(SYS) for current transfer function SYS. It can save the resulting discrete transfer function Wd(z).</td></tr>
</table>

	***poly2cof(expr,var)*** Gives the LIST of coefficients of the polynomial given in expr in the variable var. Control's Toolbox v.1.16 Author: Francesco Orabona E-mail: bremen79@infinito.it Home:http://web.genie.it/utenti/b/bremen79/
	***rts2poly(roots)*** Builds the polynomial with roots assigned as LIST. Author: Chadd L. Easterday Email: easterday@mindspring.com
	***laplace(f(t))*** Performs Laplace Transformation of f(t) Needs the tool LZT to perform symbolic calculation. LZT r7 Author: Jiri Bazant Email: georger@razdva.cz Home: http://www.razdva.cz/georger/
	***ilaplace(F(s))*** Performs Inverse Laplace Transformation of F(s). Needs the tool LZT to perform symbolic calculation. LZT r7 Author: Jiri Bazant Email: georger@razdva.cz Home: http://www.razdva.cz/georger/
	***zeta(f(k))*** Performs Zeta Transformation of f(k) Needs the tool LZT to perform symbolic calculation. LZT r7 Author: Jiri Bazant Email: georger@razdva.cz Home: http://www.razdva.cz/georger/

	***izeta(F(z))*** Performs Inverse Zeta Transformation of F(z). Needs the tool LZT to perform symbolic calculation. LZT r7 Author: Jiri Bazant Email: georger@razdva.cz Home: http://www.razdva.cz/georger/

Notes.

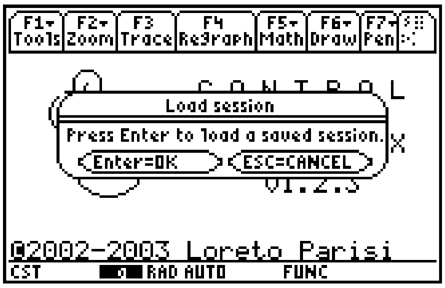
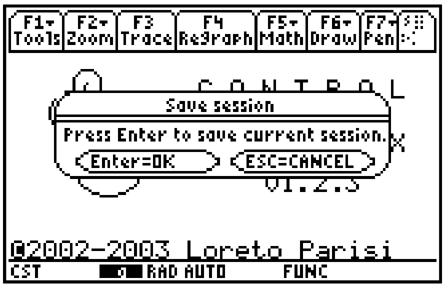
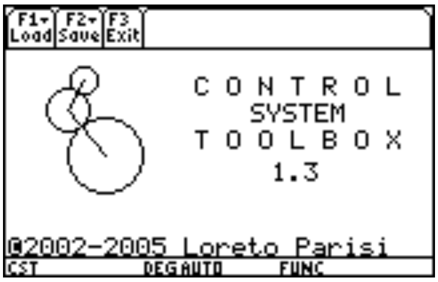
[1] About *spectre(A)*

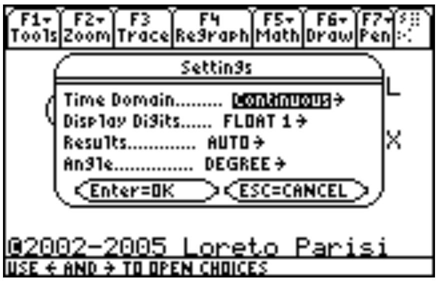
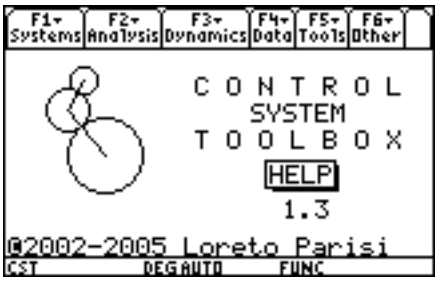
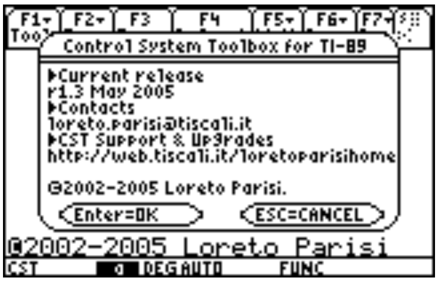
The spectral decomposition of matrix A is

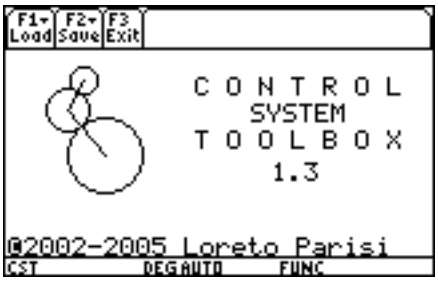
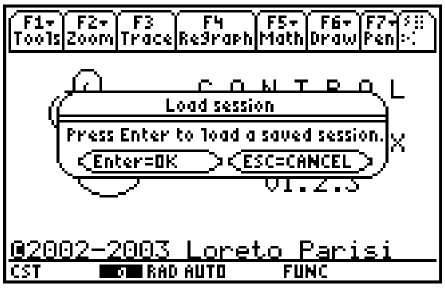
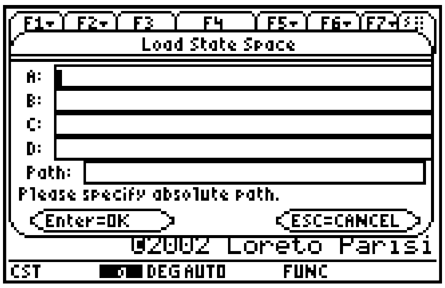
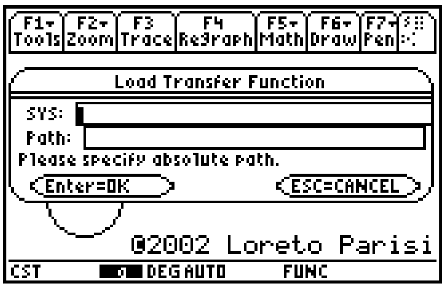
$$e^{At} = \sum_{i=1}^{\mu} u_i e^{\lambda_i t} v_i^T + \sum_{h=1}^{\upsilon} (u_{ha} \quad u_{hb}) e^{\alpha_h t} \begin{pmatrix} \cos\omega_h t & \sin\omega_h t \\ -\sin\omega_h t & \cos\omega_h t \end{pmatrix} \begin{pmatrix} v_{ha}{}^T \\ v_{hb}{}^T \end{pmatrix} \quad \text{(continuous)}$$

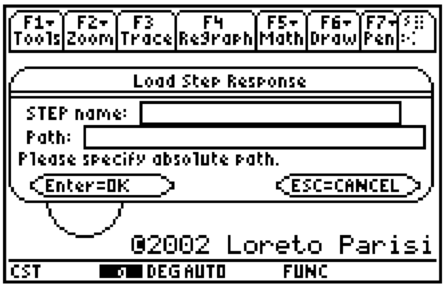
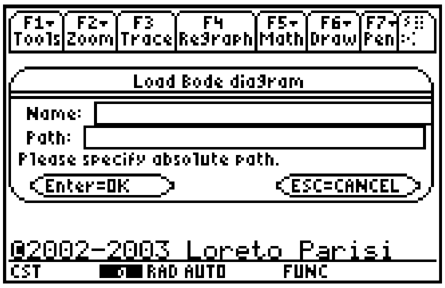
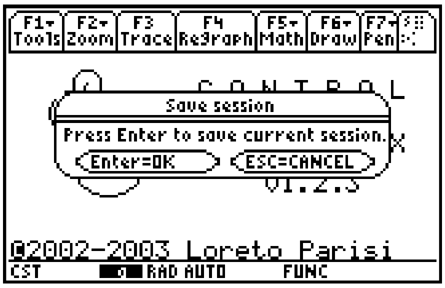
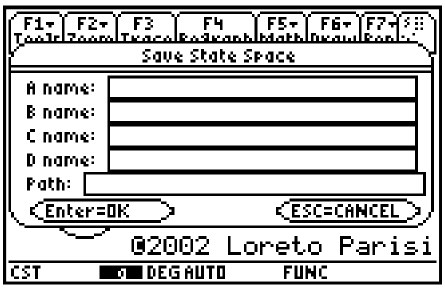
$$A^k = \sum_{i=1}^{\mu} u_i \lambda_i^k v_i^T + \sum_{h=1}^{\upsilon} (u_{ha} \quad u_{hb}) \rho_h{}^k \begin{pmatrix} \cos\theta_h k & \sin\theta_h k \\ -\sin\theta_h k & \cos\theta_h k \end{pmatrix} \begin{pmatrix} v_{ha}{}^T \\ v_{hb}{}^T \end{pmatrix} \quad \text{(discrete)}$$

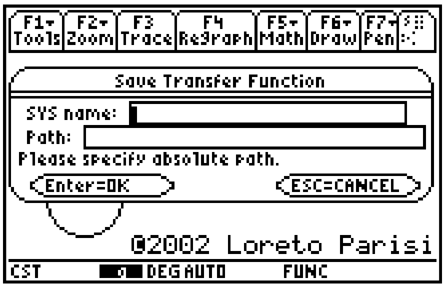
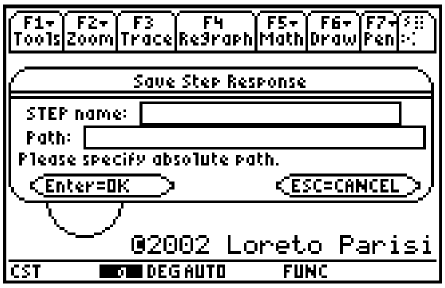
relating to real μ eigenvalues λ_i and 2υ complex eigenvalues $\lambda_h = \alpha_h \pm j\omega_h = \rho_h e^{\pm j\theta_h}$ and the relating eigenvector u_i and $u_h = u_{ha} \pm u_{hb}$.

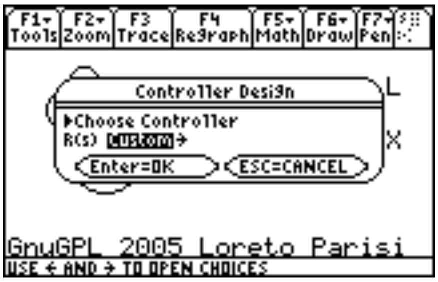
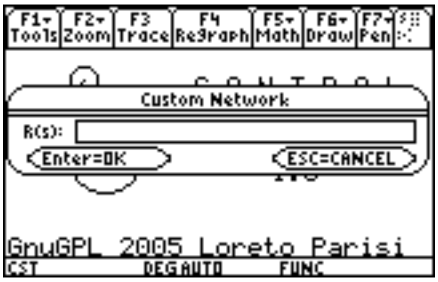
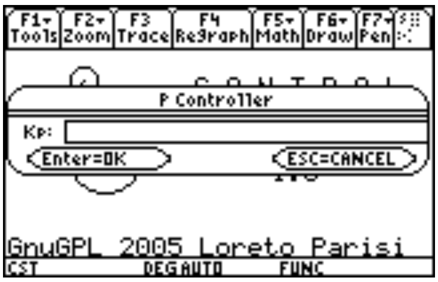
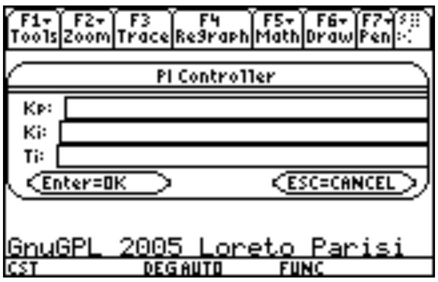
Other	
	The ***Other*** menu (F6) gives tools to manage files, the current working session, the Settings, to access to on-line help tool with help(), some information about CST, and the way to exit CST.
	***Quick Load*** Loads the current working session (i.e. transfer functions W(s) and W(z),State space, w_1(t), Tc, step response parameters,etc.) previously saved. It overwrites all the existing values for the current session. Be careful.
	***Quick Save*** Saves the current working session (i.e. transfer functions W(s) and W(z),State space, w_1(t), Tc, step response parameters,etc.).
	***File…*** The File toolbox gives access to the File & Session Management. Here you can load and save the current working session, the State Space, the Transfer Function, the Step Response and bode diagram obtained with bodex(SYS). There are three menus Load, Save and Exit. Exit menu (F3) brings to the previous toolbox.

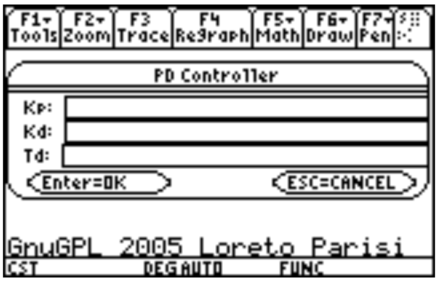
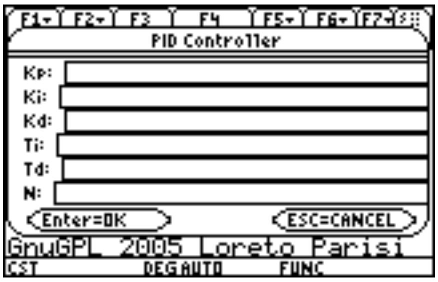
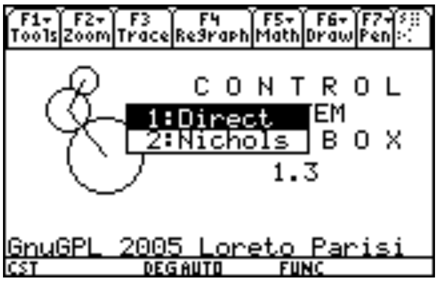
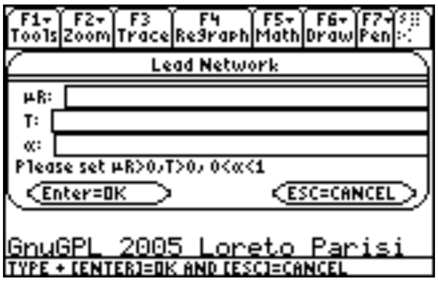
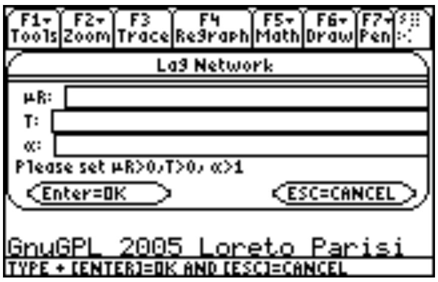
	***Settings*** It permits to modify some settings of the calculator, such as the display digits, the angle, the format of results and to switch the current Time Domain: Continuous to work with continuous time model W(s) or Discrete to work with discrete time model W(z) in the same working session.
	***help()*** Starts the help tool. To get help, simply choose a function from one of the menus and you'll get some information about it.
	***About*** Gives the current version of CST for TI-89, the way to contact the author and to obtain support and upgrades.
	***Exit*** To close Control System Toolbox for TI-89. All previous settings of the calculator will be restored. Prompts for non-saved working session.

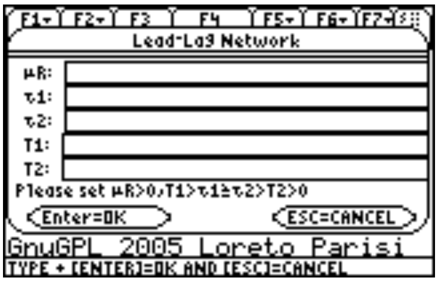
File	
	The ***File*** toolbox gives access to the File & Session Management. Here you can load and save the current working session, the State Space, the Transfer Function, the Step Response and bode diagram obtained with bodex(SYS). There are three menus Load, Save and Exit. Exit menu (F3) brings to the previous toolbox.
	***Load*** The Load menu (F1) permits to load the current working session, the State Space, Transfer Function, Step Responde and bode diagrams from the specified path.
	***Load Session*** Loads the current working session (i.e. transfer functions W(s) and W(z),State space, w_1(t), Tc, step response parameters,etc.) previously saved. It overwrites all the existing values for the current session. Be careful.
	***Load State Space*** To load state space matrixes A,B,C,D from specified path. Please use absolute path. For example, if your dynamic matrix A is stored in main as dyn, you have to input dyn in A input field and main as path. All matrixes should be in the same path.
	***Load Transfer Function*** Permits to load Transfer Function from specified path.

	***Load Step Response*** Permits to load the Step Response from specified path.
	***Load Bode diagram*** This tools permits to load a picture stored in CST folder. It's aid is in displaying Bode plots, created with bodex() first, and estimating the diagrams in a assigned frequency ω_0.
	***Save*** The Save menu (F2) permits to save the current working session, the State Space, Transfer Function, Step Responde into a specified path.
	***Save session*** Saves the current working session (i.e. transfer functions W(s) and W(z),State space, w_1(t), Tc, step response parameters,etc.).
	***Save State Space.*** Permits to save current State Space into the specified path, using given names.

	***Save Transfer Function.*** To save current transfer function into the specified path, using given name. The current SYS results from Data menu (F4).
	***Save Step Response.*** To save currrent step response into the specified path, using give name. The current step results from Data menu (F4).

Controller	
	The ***Controller*** menu (F1) is intented to design and tuning the control system.
	The ***Controller Design*** wizard will guide you throught the full design of the network's controller. First step is to choose the *network structure* from the following types: Custom (i.e. user defined), P, PI, PD, PID, Lead, Lag and Lead-Lag networks.
	***Custom Network*** Defines your own custom network's controller R(s).
	***P Controller*** Defines a proportional controller R(s) = Kp.
	***PI Controller*** Defines a PI controller R(s) as you give Kp and Ki or Kp and Ti: $R_{PI}(s) = \frac{K_P s + K_I}{s} = K_P \frac{1 + T_I s}{T_I s}$ where $T_I = \frac{K_P}{K_I}$

	***PD Controller*** Defines a PD controller R(s) as you give Kp and Kd or Kp and Td: $R_{PD}(s)=K_P+K_D s=K_P(1+T_D s)$ where $T_D=\frac{K_D}{K_P}$
	***PID Controller*** Defines a standard PID controller as you give Kp, Ki, Kd or Kp, Ti, Td: $R_{PID}(s)=\frac{K_D s^2+K_P s+K_I}{s}=K_P\frac{T_I T_D s^2+T_I s+1}{T_I s}$ or a real PID controller specifing N: $R_{PID}(s)=K_P+\frac{K_I}{s}+\frac{K_D s}{1+\frac{K_D}{K_P N}s}=K_P(1+\frac{1}{T_I s}+\frac{T_D}{1+\frac{T_D}{N}s}s)$
	***Direct Design*** Defines a Lead, Lag or Lead-lag network directly from transfer function's gain μ_R, time constant T and α parameter.
	***Lead Network*** Defines a lead network R(s) as you give the gain μ_R, time constant T and parameter α: $R(s)=\mu_R\frac{1+Ts}{1+\alpha Ts}$ Must be: $\mu_R>0,T>0,0<\alpha<1$ Usually, α=0.1 and $T=\frac{1}{\omega_c}$
	***Lag Network*** Defines a lag network R(s) as you give the gain μ_R, time constant T and parameter α: $R(s)=\mu_R\frac{1+Ts}{1+\alpha Ts}$ Must be: $\mu_R>0,T>0,\alpha>1$ Usually, $T>\frac{1}{\omega_C}$ $(T=\frac{10}{\omega_C})$

<table>
<tr>
<td>


</td>
<td>

Lead-Lag Network

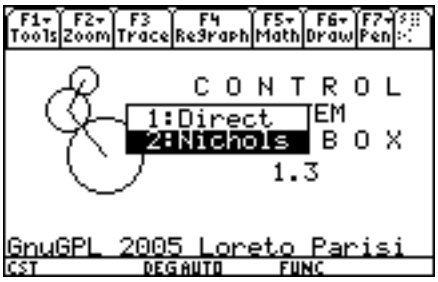
Defines a lead-lag network R(s) as you give the gain μ_R, and time constants τ_1, τ_2, T_1, T_2:

$$R(s) = \mu_R \frac{(1+\tau_1 s)(1+\tau_2 s)}{(1+T_1 s)(1+T_2 s)}$$

Must be:

$\mu_R > 0, T_1 > \tau_1 \geq \tau_2 > T_2 > 0$

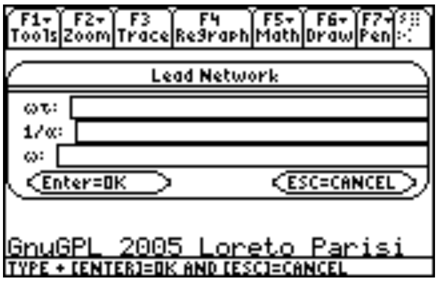
Usually, $T_1 T_2 = \tau_1 \tau_2, \tau_2 > \frac{1}{\omega_C} > T_2$

</td>
</tr>
<tr>
<td>


</td>
<td>

Nichols Design

Defines a Lead, Lag or Lead-lag network using Nichols's standard networks parameters $\omega\tau$, $1/\alpha$ at ω_0. In most cases ω_0 will be the critical frequency ω_C.

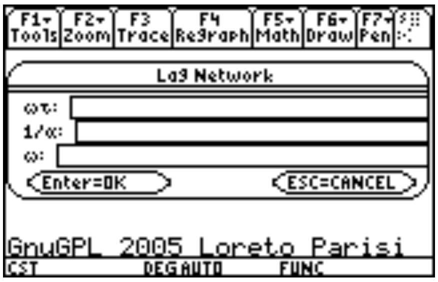
</td>
</tr>
<tr>
<td>


</td>
<td>

Lead Network

Defines a Lead network using Nichols's standard networks parameters $\omega\tau$, $1/\alpha$ at ω_0.

$$R(s) = \frac{1+s\tau}{1+s\alpha\tau}$$

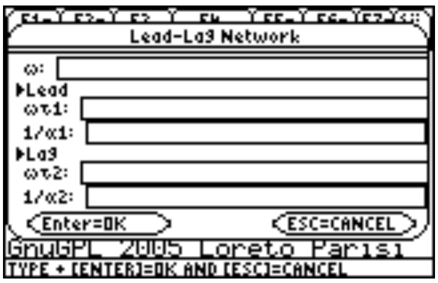
</td>
</tr>
<tr>
<td>


</td>
<td>

Lag Network

Defines a Lag network using Nichols's standard networks parameters $\omega\tau$, $1/\alpha$ at ω_0.

$$R(s) = \frac{1+s\alpha\tau}{1+s\tau}$$

</td>
</tr>
<tr>
<td>


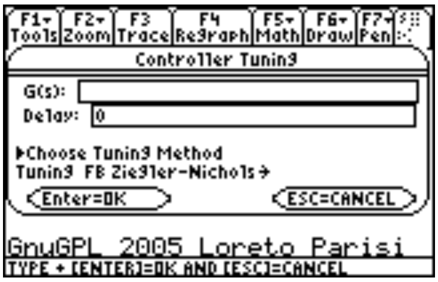
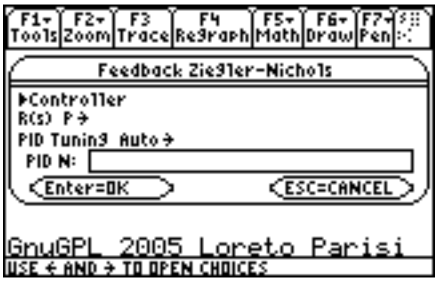
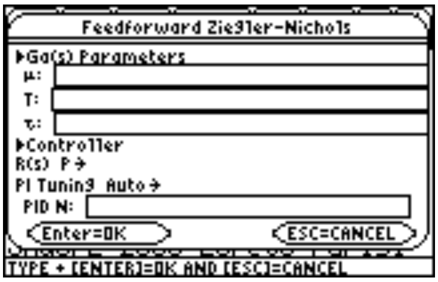
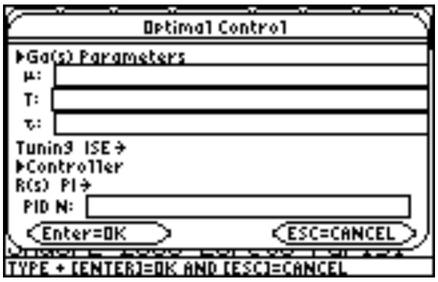
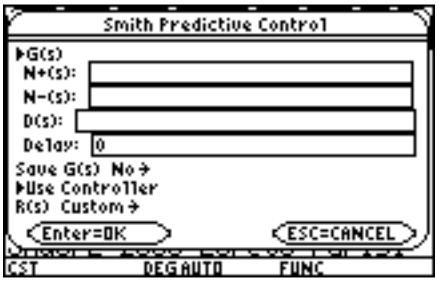
</td>
<td>

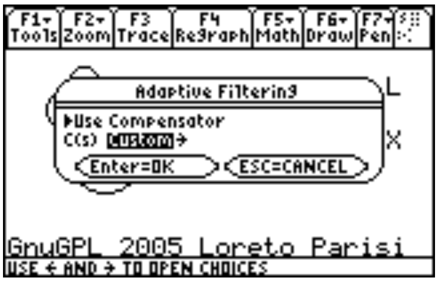
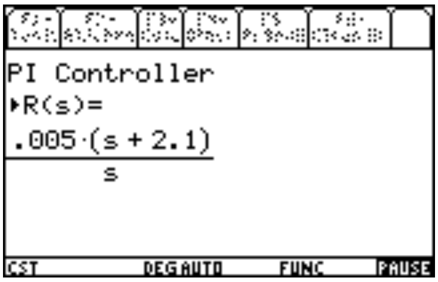
Lead-Lag Network

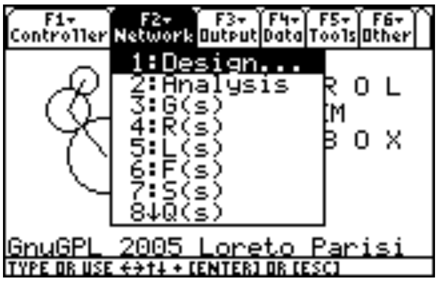
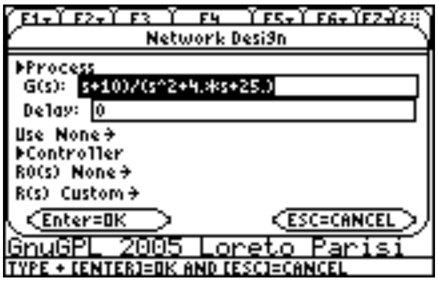
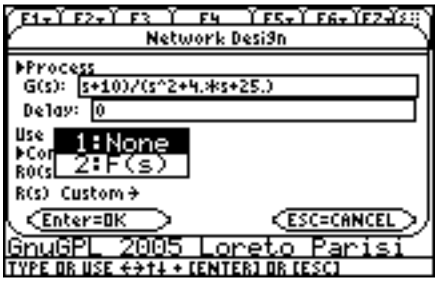
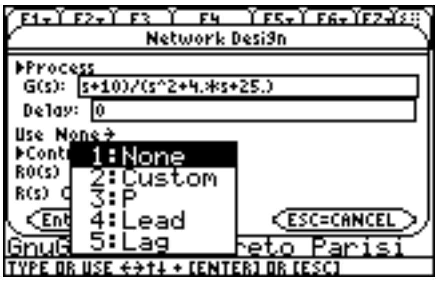
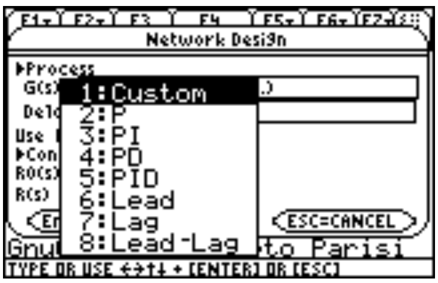
Defines a Lead-Lag network using Nichols's standard networks parameters $\omega\tau_1$, $1/\alpha_1$ for the lead and $\omega\tau_2$, $1/\alpha_2$ for the lag network at frequency ω_0.

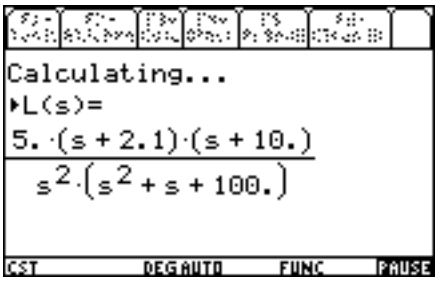
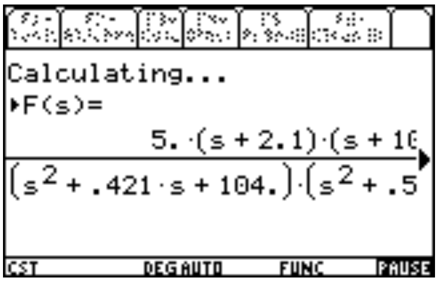
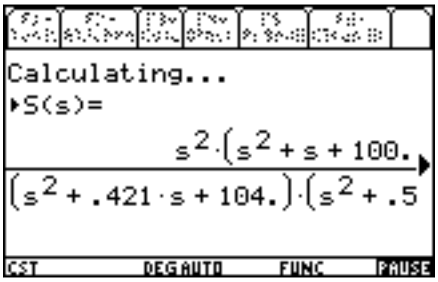
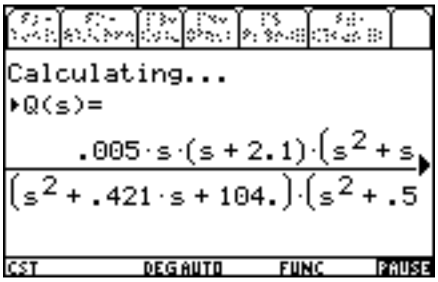
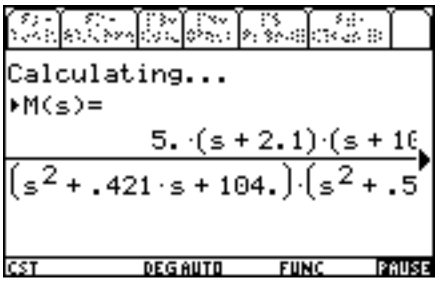
$$R(s) = \frac{1+s\tau_1}{1+s\alpha_1\tau_1}\frac{1+s\alpha_2\tau_2}{1+s\tau_2}$$

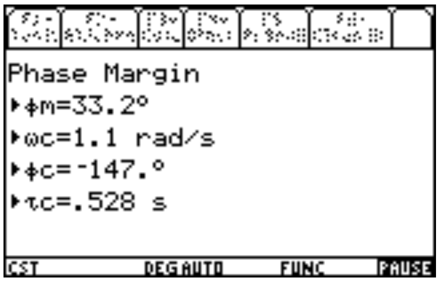
</td>
</tr>
</table>

	***Controller Tuning*** The Controller Tuning wizard will guide you through the tuning of the network's controller R(s) for the given process G(s) and its delay. First choose the tuning method from ones avaiable: *Feedback Ziegler-Nichols*, *Feedforward Ziegler-Nichols*, *Optimal Control*, *Predictive Control* and *Adaptive Filtering.*
	***Feedback Ziegler-Nichols*** Uses the *Closed Loop Ziegler-Nichols* method to tune the controller for the feedback network. Choose the desidered structure for R(s) – P, PI or PID. Only for PIDs, choose the assignment method for gain and phase margins (Auto, assign Gain Margin or assign Phase Margin) and the parameter N if you wish to use a real PID controller, instead of a standard PID controller.
	***Feedforward Ziegler-Nichols*** Uses the *Open Loop Ziegler-Nichols* method to tune the controller for the approximate process (obtained from the step response using the *areas method*) $G_a(s) = \frac{\mu}{1+Ts}e^{-\tau s}$. Choose the structure (P, PI or PID) for R(s) and for PIs only the assignment method for the phase margin (Auto, assign Phase Margin).
	***Optimal Control*** Uses optimization methods to tune the controller R(s): ISE (*Integral Square Error*), ISTE (*Integral Square Time Error*) and IST^{2}E (*Integral Square Time^2 Error*). Kp, Ti and Td are defined by a table as follows: $K_P = \frac{a_1}{\mu}\theta^{b_1}, T_I = \frac{T}{a_2 + b_2\theta}, T_D = a_3 T\theta^{b_3}$
	***Smith Predictive Control*** Used to tune network's controllers for processes with postive real zeros or time delays. The process G(s) is given as $G(s) = \frac{N^-(s)N^+(s)}{D(s)}e^{-\tau s}$. The predictor P(s) and the network transfer function L'(s) for the given controller R(s) are $P(s) = \left(1 - \frac{N^+(s)}{N^+(-s)}e^{-\tau s}\right)\frac{N^-(s)N^+(-s)}{D(s)}$ and $L'(s) = (G(s) + P(s))R(s)$.

<table>
<tr>
<td>
</td>
<td>Adaptive Filtering
Uses a pre-filtering technique (compensation of input signal) to improve static and dynamic behaviour. You have to choose the structure for the compensator C(s). We suppose you have defined it as a controller (custom, lead, lag or lead-lag) yet.</td>
</tr>
<tr>
<td>
</td>
<td>Custom, P, PI, PD, PID, Lead, Lag, Lead-Lag
Shows the controller defined for that structure. Note that you have to choose the controller first to perform the analysis, but it’s possible to define (design or tuning) more controllers, then choose one of them as the current R(s).</td>
</tr>
</table>

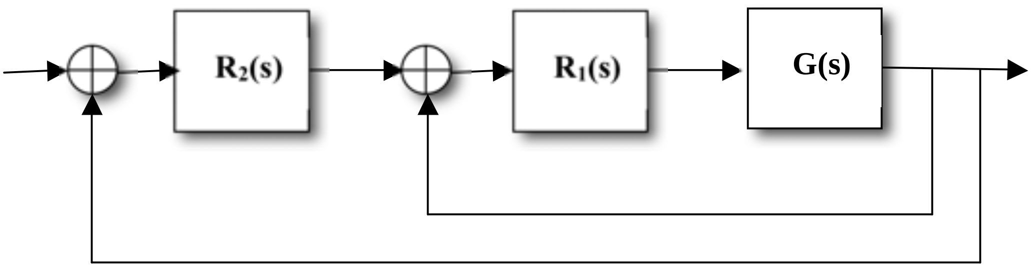
Network	
	The ***Network** menu (F2)* permits to design and analyse the control system, calculating gain and phase margins, and the network transfer functions.
	***Network Design*** Define the process G(s) and its delay, the controllers R_0(s) and R(s).
	***Network Design*** Use the inner loop transfer function F(s) as G(s) in the unstable processes control systems. Now you can tune the controller against the inner closed loop transfer function. See notes for more informations.
	***Network Design*** Choose the controller R_0(s) between custom, proportional, lead or lag structures. Generally, use it to satisfy static requirements.
	***Network Design*** Choose the controller R(s) between all structures avaiable to satisfy dynamic requirements.

 	***L(s)*** The network tranfer function $L(s) = R(s)G(s)$
 	***F(s)*** The network transfer function $F(s) = \frac{R(s)G(s)}{1+R(s)G(s)}$
 	***S(s)*** The network transfer function $S(s) = \frac{1}{1+R(s)G(s)}$
 	***Q(s)*** The network transfer function $Q(s) = \frac{R(s)}{1+R(s)G(s)} = F(s)G(s)^{-1} = R(s)S(s)$
 	***M(s)*** The network transfer function $M(s) = G(s)S(s)$

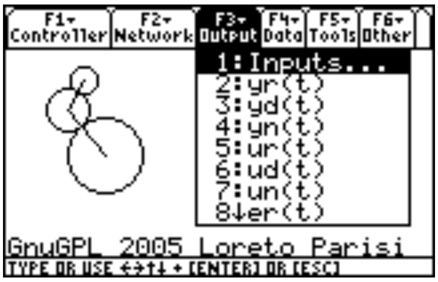
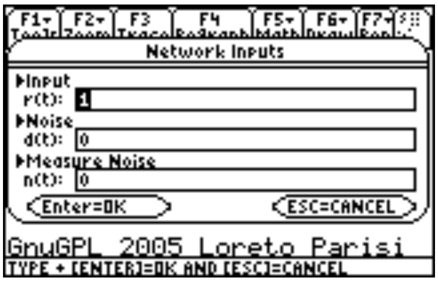
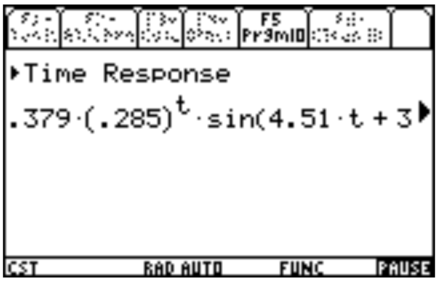
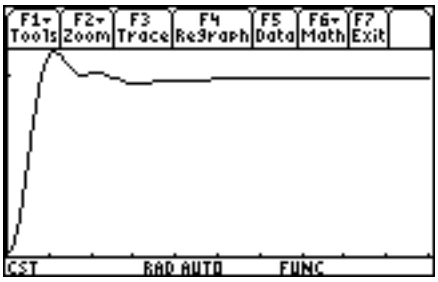
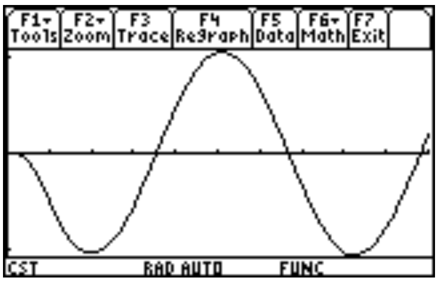
<table>
<tr>
<td>


</td>
<td>Analysis
Performs an analysis of the defined control system, calculating the Gain and the Phase margins.</td>
</tr>
</table>

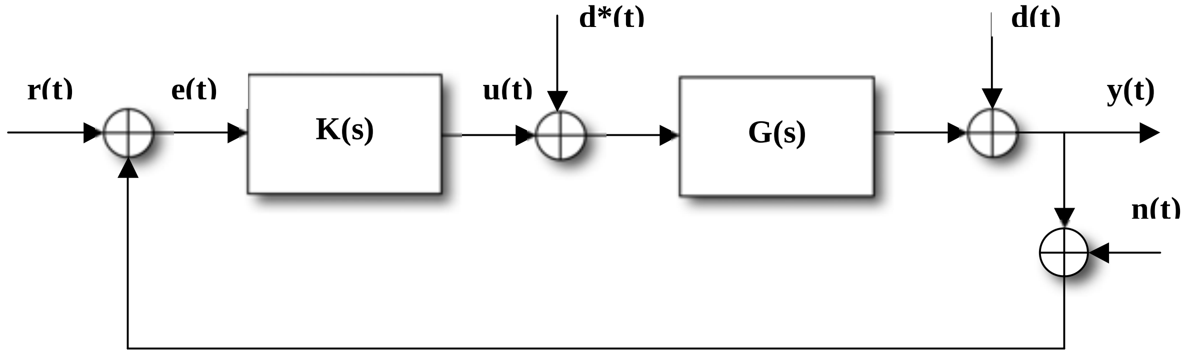
Notes.
About *Network Design*.
1. When G(s) is such an unstable process, we'll use a block model with a inner control loop, like the block diagram below.



And we'll tune $R_1(s)$ to stabilize G(s) and $R_2(s)$ against $F(s) = \frac{R_1(s)G(s)}{1 + R_1(s)G(s)}$ to satisfy given requirements. To do so, first design $R_1(s)$ as usual. When the inner closed loop is stable, you can design $R_2(s)$ choosing in *Network Design* **Use F(s)** as new G(s) from the drop down menu.

Output	
	The **Output** menu (F3) permits to perform a time domain analisys of the closed loop system against inputs, noises and measure noise.
	**Inputs** To set the input r(t), noise inputs d(t) and d*(t), and measure noise input n(t). To set a step input of amplitude K use K for r(t). To set a sinusoidal measure noise, use sin(ωt) as n(t).
	**yr(t), yd(t),yn(t)** The output y time response against input r(t), noise input d(t) and measure noise input n(t).
	**ur(t), ud(t), un(t)** The control variable u time response against input r(t), noise input d(t) and measure noise input n(t).
	**er(t), ed(t), en(t)** The error e=r-y time response against input r(t), noise input d(t) and measure noise input n(t).

Notes.
About *Output* menu.
1. We assume a Closed Loop Control System block model like that below.



2. We assume the following transfer functions.

$$\begin{bmatrix} Y(s) \\ U(s) \\ E(s) \end{bmatrix} = \begin{bmatrix} F(s) & S(s) & -F(s) \\ Q(s) & -Q(s) & -Q(s) \\ S(s) & -S(s) & F(s) \end{bmatrix} \cdot \begin{bmatrix} R(s) \\ D(s) \\ N(s) \end{bmatrix}$$

where

$$F(s) = \frac{K(s)G(s)}{1+K(s)G(s)},$$

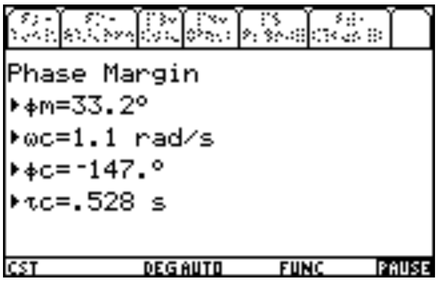
$$S(s) = \frac{1}{1+K(s)G(s)},$$

$$Q(s) = \frac{K(s)}{1+K(s)G(s)},$$

and $Y^*(s)=M(s)D^*(s)$ where $M(s)=G(s)\,S(s)$.

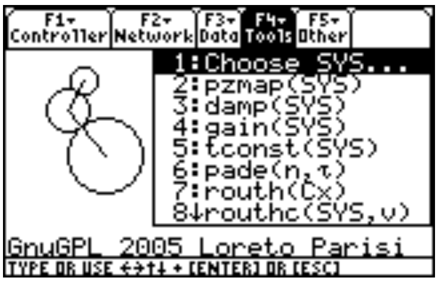
Data

The ***Data*** menu (F3) shows detailed informations about the control system.

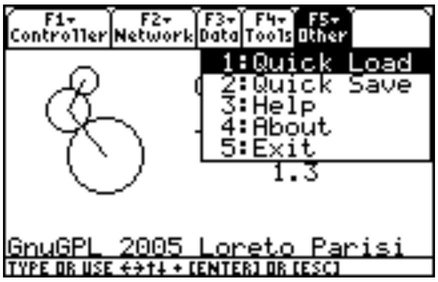
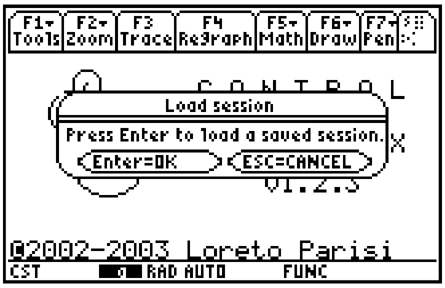
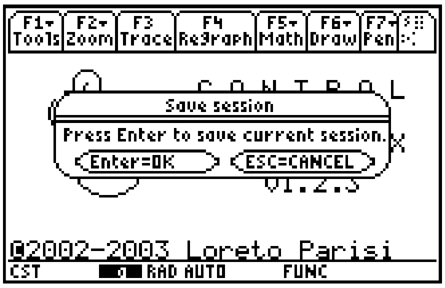
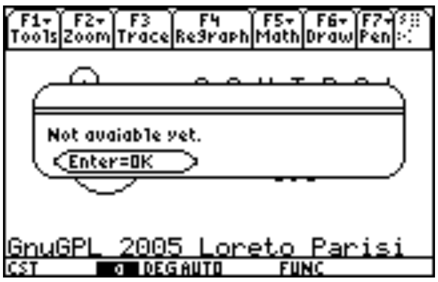


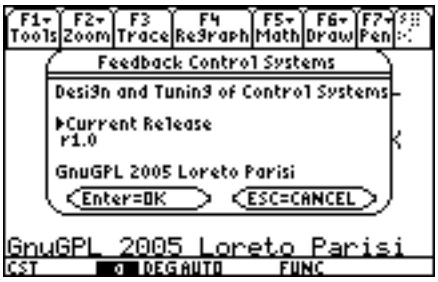
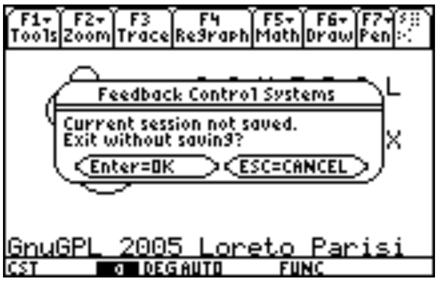
Margins
Shows the Gain and Phase margins of the control system.

Tools



The ***Tools*** menu (F4) permits you to perform a detailed analysis of transfer function SYS. You need to choose the current transfer fuction SYS first.

Other	
	The ***Other*** menu (F3) shows some help, info and permits to exit the program.
	***Quick Load*** Loads the current working session (i.e. transfer functions of process, network and controllers, calculated margins, etc.) previously saved. It overwrites all the existing values for the current session. Be careful.
	***Quick Save*** Saves the current working session (i.e. transfer functions of process, network and controllers, calculated margins, etc.).
	***Help*** Shows some help.

	***About*** Shows some info.
	***Exit*** Exit the program. Unsaved session will be losed.