

AnT 4.669

Release 2a

Examples.
Phase locked loop

University of Stuttgart
Nonlinear Dynamics Group
©2000 – 2003

April 8, 2003

Chapter 1

1.1 Definition of the system

The phase locked loop is a dynamical system continuous in time with time delay, defined by:

$$\dot{x}(t) = -R \sin(x(t - \tau)) \quad (1.1)$$

For the investigation within the AnT package the system function 1.1 has to be implemented according to the interface for delay differential equations¹ and connected to the simulator using the `DDE_Proxy`, as follows:

¹See Reference Manual, Part II

```
#include "AnT.hpp"

#define R    parameters[0]
#define Xt   delayState[0]

bool delayed_pll (const Array<real_t>& currentState,
                  const Array<real_t>& delayState,
                  const Array<real_t>& parameters,
                  Array<real_t>& rhs)
{
    rhs[0] = -R * sin (Xt);
    return true;
}

#undef R
#undef Xt

extern "C" {

void connectSystem ()
{
    DDE_Proxy::systemFunction = delayed_pll;
}

}
```

The part of the initialization file, needed for the simulation of the phase locked loop, concerning the dynamical system itself and its simulation, can look like the following:

```
dynamical_system =
{
  type = "dde",
  name = "pll",
  delay = 1

  state_space_dimension = 1,
  temporal_initial_function[0] = { type = "const",
                                   const_value = 1.0
                                 },
  parameter_space_dimension = 1,
  parameters = { parameter[0] = { value = 4.157,
                                   name = "R" }
                },
  integration = { step_size = 0.001,
                 method = "rk44"
               },
  number_of_iterations = 10000,
}
```

These settings specify, that the system is a delay differential equation with one state variable and one parameter. The delay time ($\tau = 1$), the temporal initial function for the single state variable and the value of the parameter are given. Additionally the number of the integration steps, integration method (Runge–Kutta method of the order 4 without step size adaption) and the integration step size (10^{-3} time units) are set.

1.2 Initial functions

In the examples presented in this section several temporal initial functions will be tested.

1.2.1 Constant initial function

```
...
temporal_initial_function[0] = { type = "const",
                                const_value = 1.0
                                }
...
```

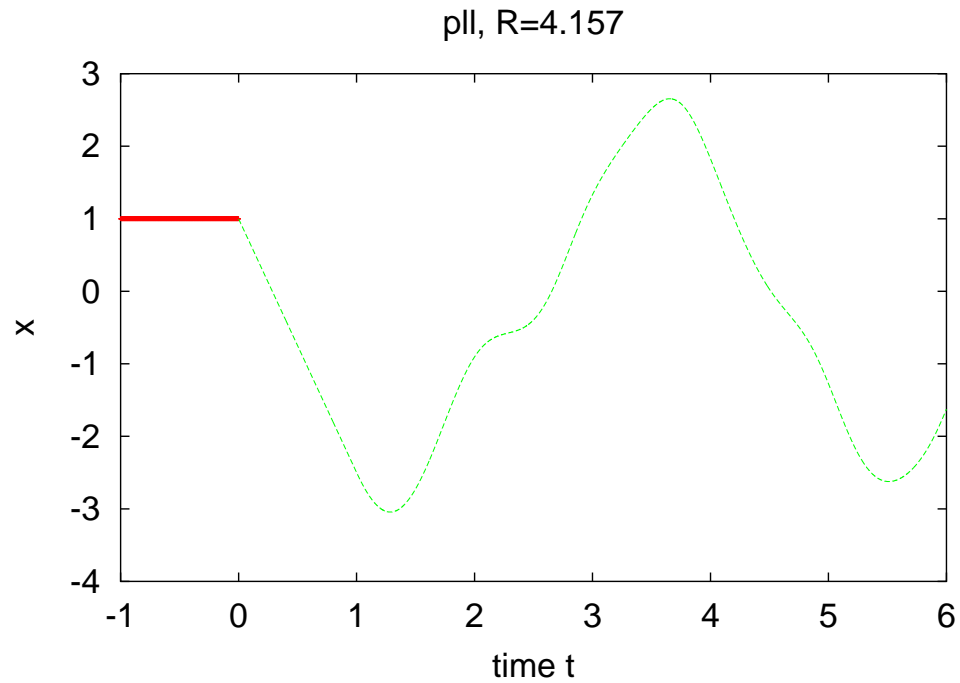


Figure 1.1:

1.2.2 Fermi initial function

```
...
temporal_initial_function[0] = {type = "fermi",
                                edge = -0.5,
                                offset = -1.0,
                                mu = 2.0,
                                sigma = 0.025
                                }
...
```

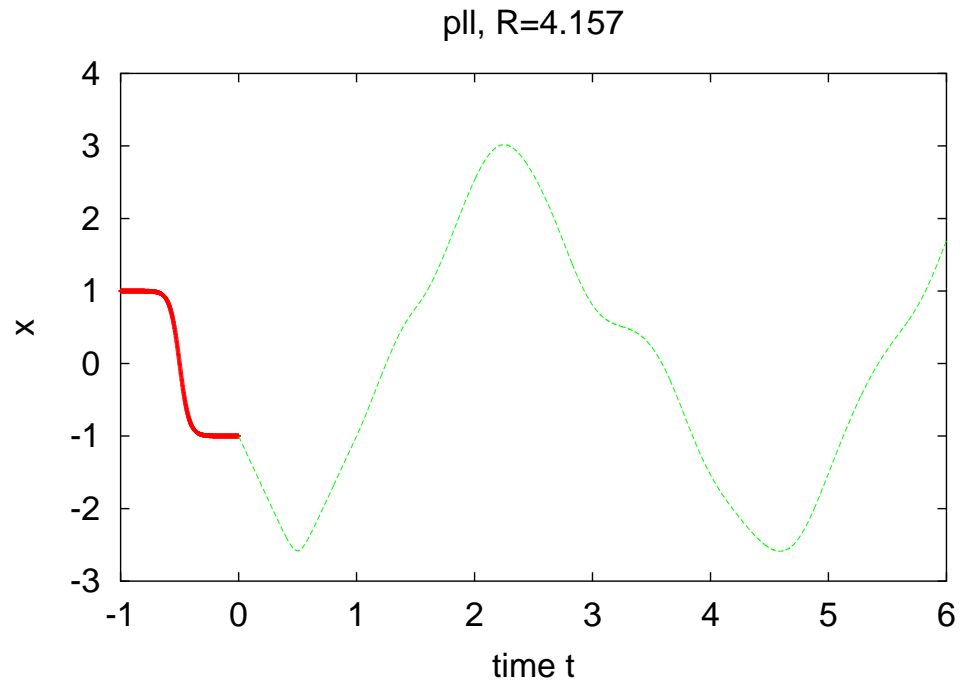


Figure 1.2:

1.2.3 Gauss initial function

```
...
temporal_initial_function[0] = {type = "gauss",
                                amplitude = 2.0,
                                offset = 0.5,
                                mu = -0.5,
                                sigma = 0.15
                                }
...
```

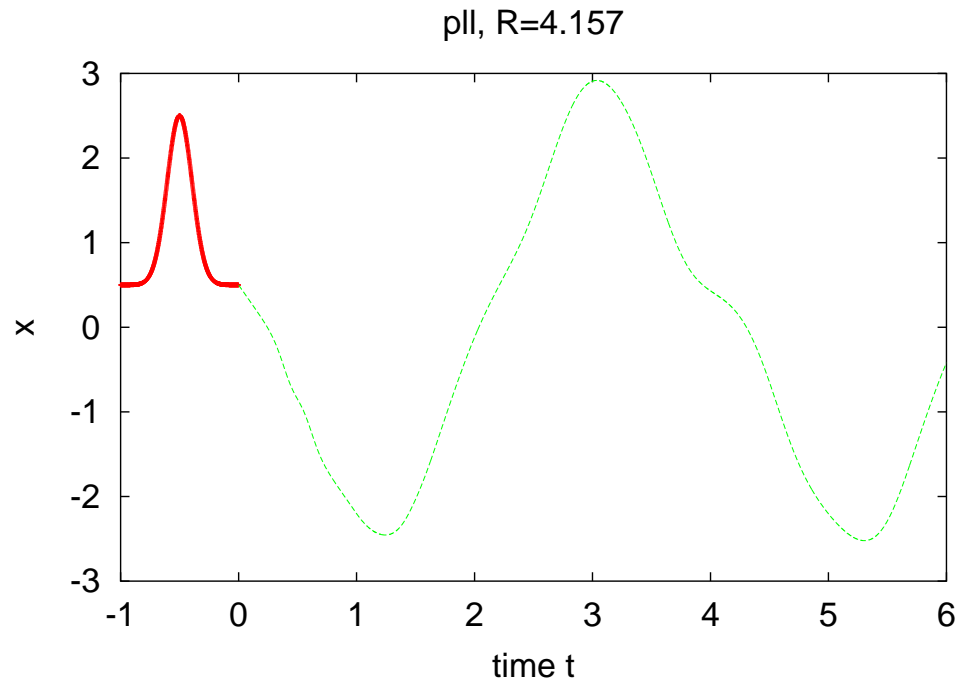


Figure 1.3:

1.2.4 Linear initial function

```
...
temporal_initial_function[0] = {type = "linear",
                                slope = 0.5,
                                offset = 1.0
                                }
...
```

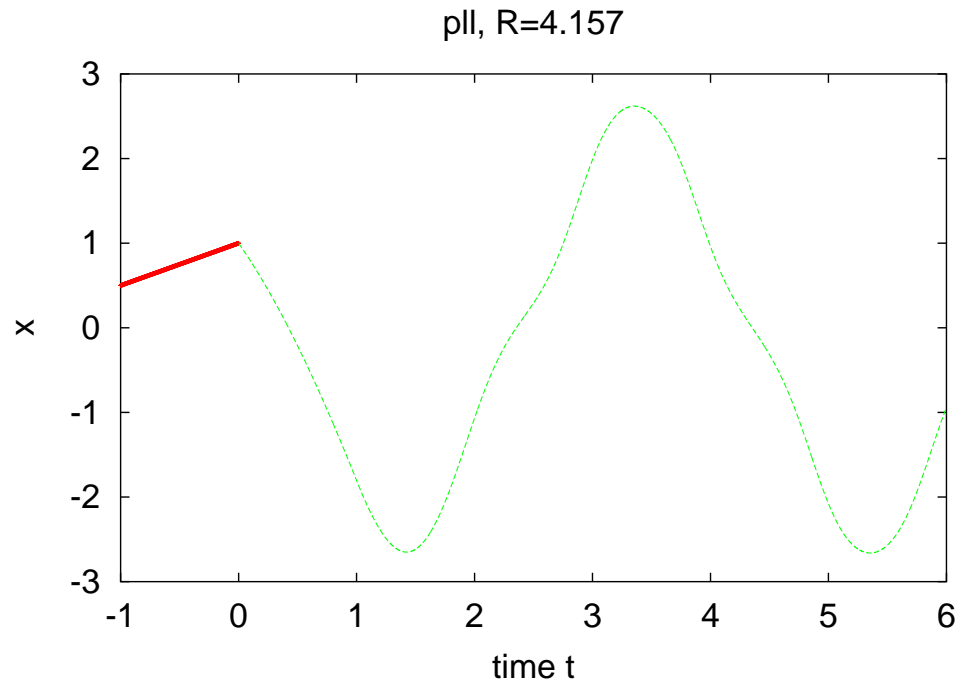


Figure 1.4:

1.2.5 Sinus initial function

```
...
temporal_initial_function[0] = {type = "sin",
                                amplitude = 1.0,
                                frequency = 2,
                                offset = 0.5,
                                phase = 0.0
                                }
...
```

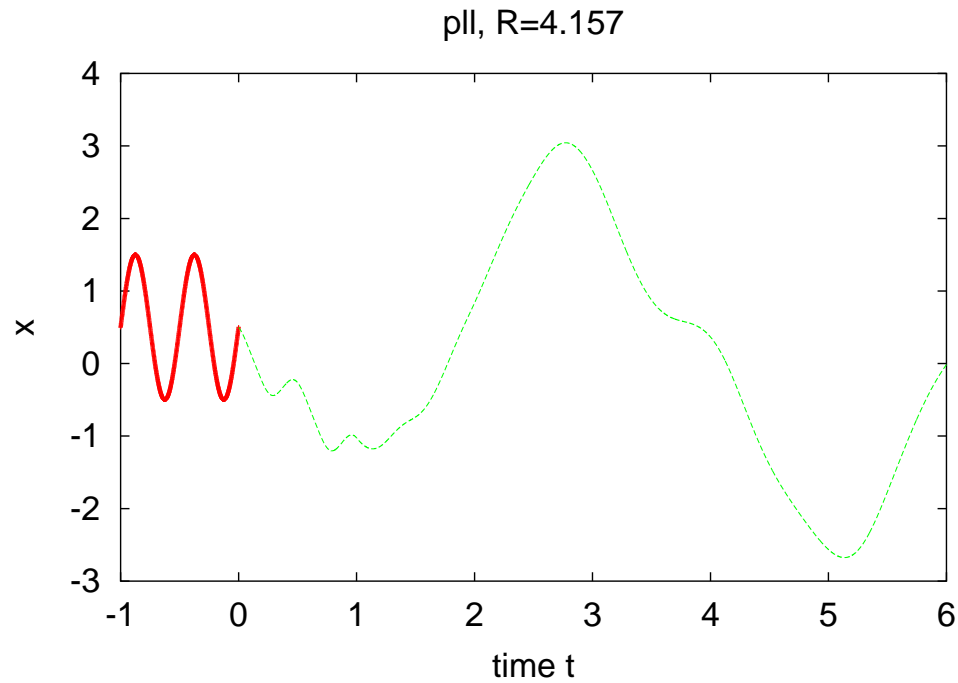


Figure 1.5:

1.2.6 Sinc initial function

```
...
temporal_initial_function[0] = {type = "sinc",
                                amplitude = 2.0,
                                frequency = 4,
                                offset = 0.5,
                                mu = -0.5,
                                phase = 0.0
                                }
...
```

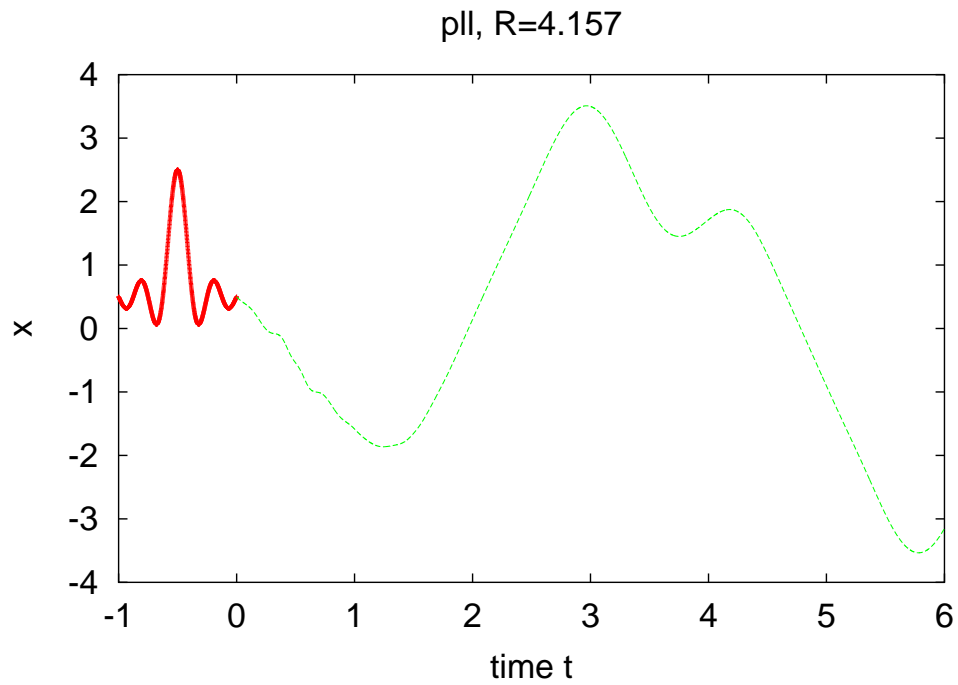


Figure 1.6:

1.2.7 Singular initial function

```
...
temporal_initial_function[0] = {type = "singular",
                                singular_value = 1.5,
                                residual_value = -0.5,
                                singular_time = -0.5
                                }
...
```

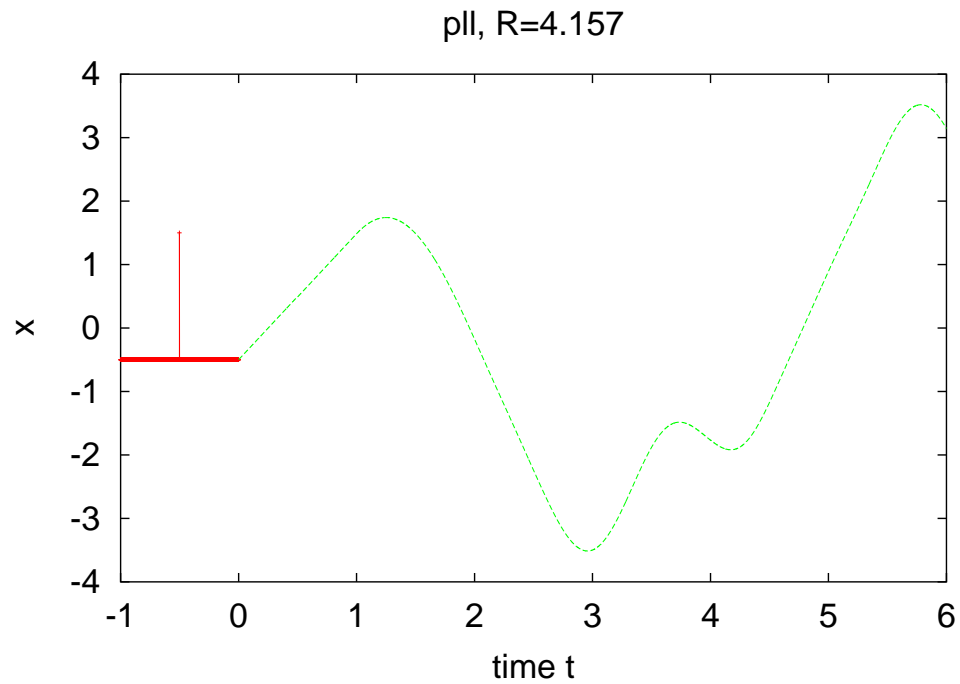


Figure 1.7:

1.2.8 Step initial function

```
...
temporal_initial_function[0] = {type = "step",
                                step_value = 1.0,
                                residual_value = 0.0,
                                first_time = -0.9,
                                second_time = -0.7
                                }
...
```

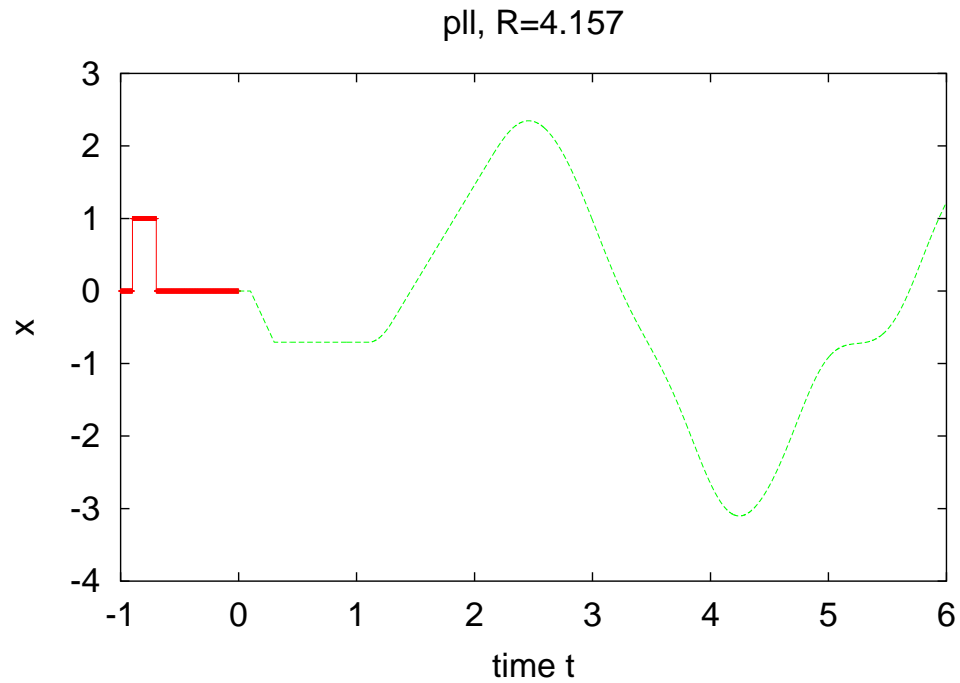


Figure 1.8: