
DTU

28451 – Optimizing Plantwide Control

Jakob K Huusom

Identification of models

Linear systems

Contents

- Linear systems
- Experimental system identification
- Linearisation of non-linear systems
- Computer example
- Wrap-up
- Computer exercise

Learning objectives

- Define linear systems
- Identify important features of linear systems behaviour
- Derive linear representations for general non-linear systems
- Use Matlab/Simulink tools for linear systems / linearisation

Reading:

- Åström and Murray, *Feedback Systems, Ch 6*
- Seborg et al., *Process Dynamics and Control*
  - *2ed. Ch. 7*
  - *3ed. Ch. 6*

Why linear systems?

Many important *tools*

- **Frequency response, step response, etc**
  - Traditional tools of control theory
  - Developed in 1930's at Bell Labs; intercontinental telecom
- **Classical control system toolbox**
  - Nyquist plots, gain/phase margin
  - Loop shaping
- **Optimal control and estimators**
  - Linear quadratic regulators
  - Kalman estimators
- **Robust control design**
  - *H_∞ control design*
  - μ analysis for structured uncertainty
- And the likes...

Linear systems

The general form of a linear state-space system:

$$\frac{dx}{dt} = Ax + Bu \quad x \in R^n, u \in R$$
$$y = Cx + Du \quad y \in R$$

where $A \in R^{n \times n}$, $B \in R^{n \times p}$, $C \in R^{q \times n}$ and $D \in R^{q \times p}$. In the special case of a single-input, single-output system, B is a column vector, C a row vector and D a scalar. *n* denotes the number of states (x), *p* the number of inputs (u), and *q* is number of measured states (y).

The expression is a system of linear first-order differential equations with input u, state x and output y.

Linear systems

A *linear system*: The output is jointly linear in the initial condition for the system and the input to the system.

$$x(0)=0,\quad u(t)=u_1(t) \qquad x(0)=0,\quad u(t)=u_2(t)$$
$$\rightarrow\quad y(t)=y_1(t) \qquad \rightarrow\quad y(t)=y_2(t)$$
$$\Downarrow$$
$$x(0)=0,\quad u(t)=\alpha u_1(t)+\beta u_2(t)$$
$$\rightarrow\quad y(t)=\alpha y_1(t)+\beta y_2(t)$$

where y_i is the output associated with the input u_i. This property is called linear *superposition*.

Linear systems - time invariance

Time invariance is an important concept used to describe a system whose properties do not change with time.

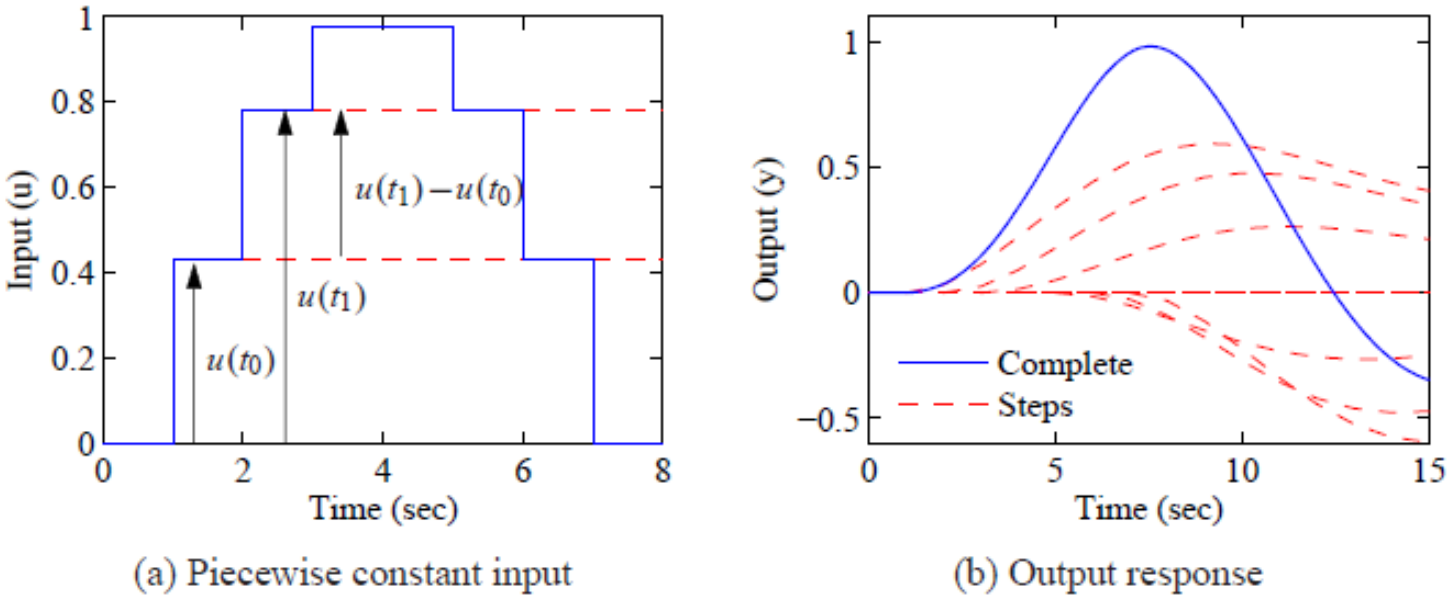
More precisely, for a time-invariant system the following holds:

$$\text{suppose} \quad u(t) \rightarrow y(t)$$
$$\textit{then}$$
$$u(t+a) \rightarrow y(t+a)$$

Systems that are linear and time-invariant, are called LTI systems!

Transfer functions or state space models with constant coefficients are examples of models for a LTI system.

Linear systems - time invariance



(a) Piecewise constant input

(b) Output response

LTI models have the interesting property that their response to an arbitrary input is completely characterized by their response to step inputs or their response to short "impulses."

Linear systems – solutions using linear algebra (matrix exponential)

A differential equation of the form

$$\frac{dx}{dt} = Ax + Bu \quad x \in R^n, u \in R$$
$$y = Cx + Du \quad y \in R$$

The general form of the solution to the linear differential equation is given by the following matrix exponential:

$$y(t) = Ce^{At}x(0) + \int_0^t Ce^{A(t-\tau)}Bu(\tau)\,d\tau + Du(\tau)$$

Which is called the *convolution equation.*

This has important and useful functions in control theory (see lecture 7 & 9):

- the dynamics of the system are characterized by the matrix A
- *A plays a* critical role in the stability and performance of the system
- The equation describes the dynamic behavior of the linear system.

Linear systems – stability

For a linear system

$$\dot{x} = Ax$$

Stability is determined by the **eigenvalues of *A*, λ_n:**

$$\dot{x} = Ax \quad \rightarrow x(t) = e^{At}x(0)$$

Stable *if* $\lambda_i \leq 0$

Asy stable *if* $\lambda_i < 0$

unstable *if* $\lambda_i > 0$

If all eigenvalues of A have strictly negative real parts: Stable!
Unstable if any eigenvalue of *A* has a strictly positive real part.

Linear system – steady-state response

The general solution of a (stable) linear system contains a transient (which decays to zero) and a steady-state region (which persists over time):

$$y(t) = Ce^{At}x(0) + \int_0^t C^{A(t-\tau)}Bu(\tau)\,d\tau + Du(\tau)$$

Response to initial conditions

Response to input

Transient response

Steady-state response

Matlab tools for linear systems

A = [-1 1; 0 -1]; B = [0; 1]; C = [1 0]; D = [0];

x0 = [1; 0.5]; t = 0:0.1:10;

u = 0.2*sin(5*t) + cos(2*t);

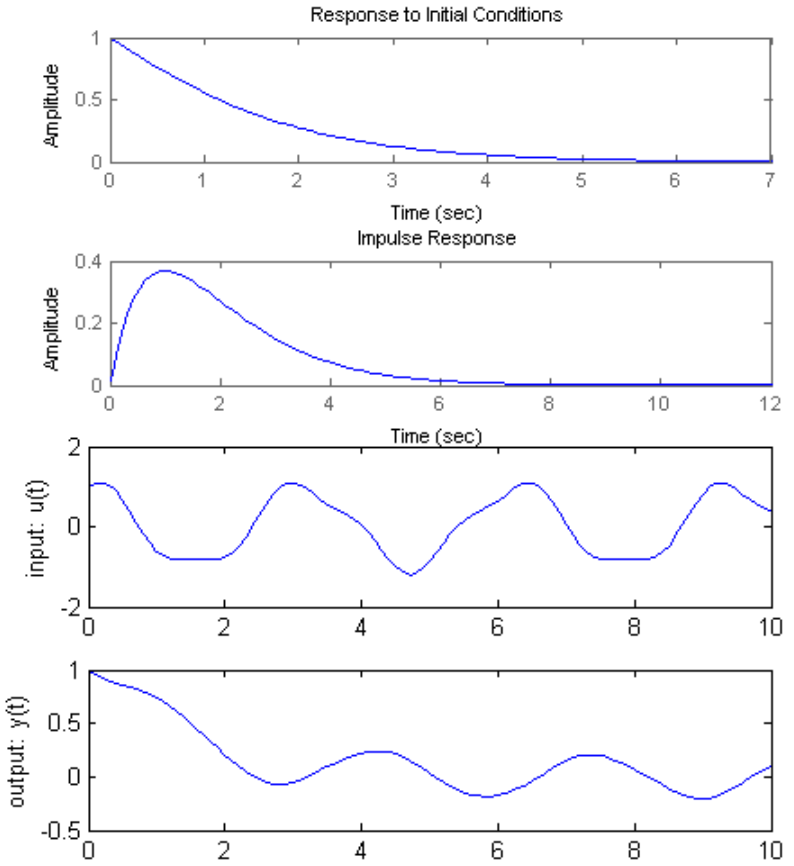
$$y(t) = Ce^{At}x(0) + \int_0^t C^{A(t-\tau)}Bu(\tau)\,d\tau + Du(\tau)$$

sys = ss(A,B,C,D);

initial(sys, x0); impulse(sys); lsim(sys, u, t, x0)

Other MATLAB commands

- gensig, square, sawtooth – produce signals of diff. types
- step, impulse, initial, lsim, ltview – time domain analysis
- bode, freqresp, evalfr – frequency domain analysis



Data-driven system identification

In industry one of the most frequently used method for achieving a linear approximation to a process (potentially non-linear) is through specific plant tests in either open or closed loop, followed by system identification of simple model structures.

Very often step tests are applied, but it could also be a more complex input signal which is aimed at improving the statistical properties of the system identification process.

Procedure:

1. Select a one or more potential model structures.
2. Experiment in open or closed loop for one input at a time
   -> collect [u Y] or [y^{set} U Y] data.
3. Estimate the unknown parameters in a given model.
4. Compare the model responses to the plant data for verification.
5. If possible validate the model fit with independent data.

Linear systems - Step response

Open loop step responses are often used for plant trials since these are very easy to interpret and allow efficient estimation of simple transfer function models.

For MIMO or MISO processes you only perturb one input at a time!

To assess whether the process is non-linear it is recommended to have both positive and negative step perturbations.

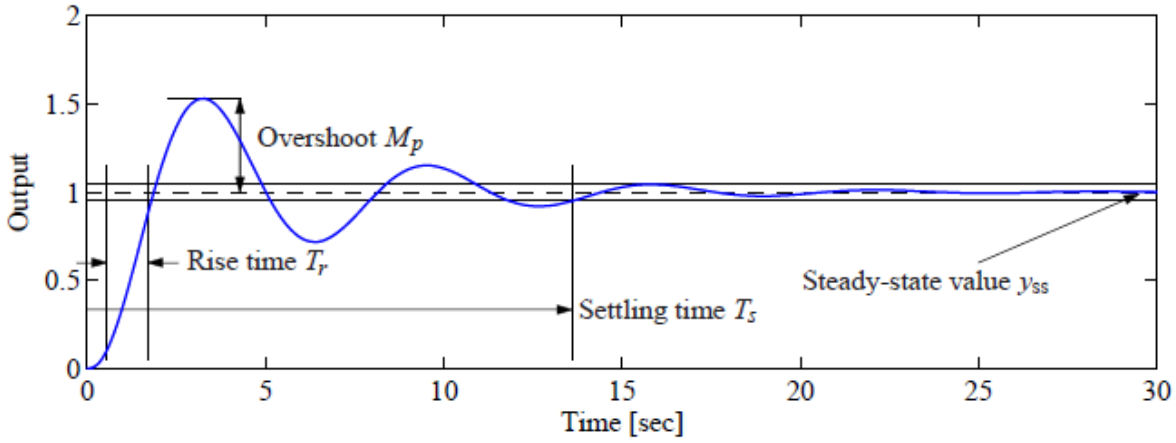
A challenge arises if the open loop system cannot be perturbed by a sufficiently large step.

- Closed loop system identification may be needed, where a stabilizing P or PI controller is implemented.
- The step response test is imposed via the set point of the controller.
- The process model can still be approximated from [U Y] data, but this data is not true step response data. Alternatively the closed loop transfer function can be identified, the process consecutively.

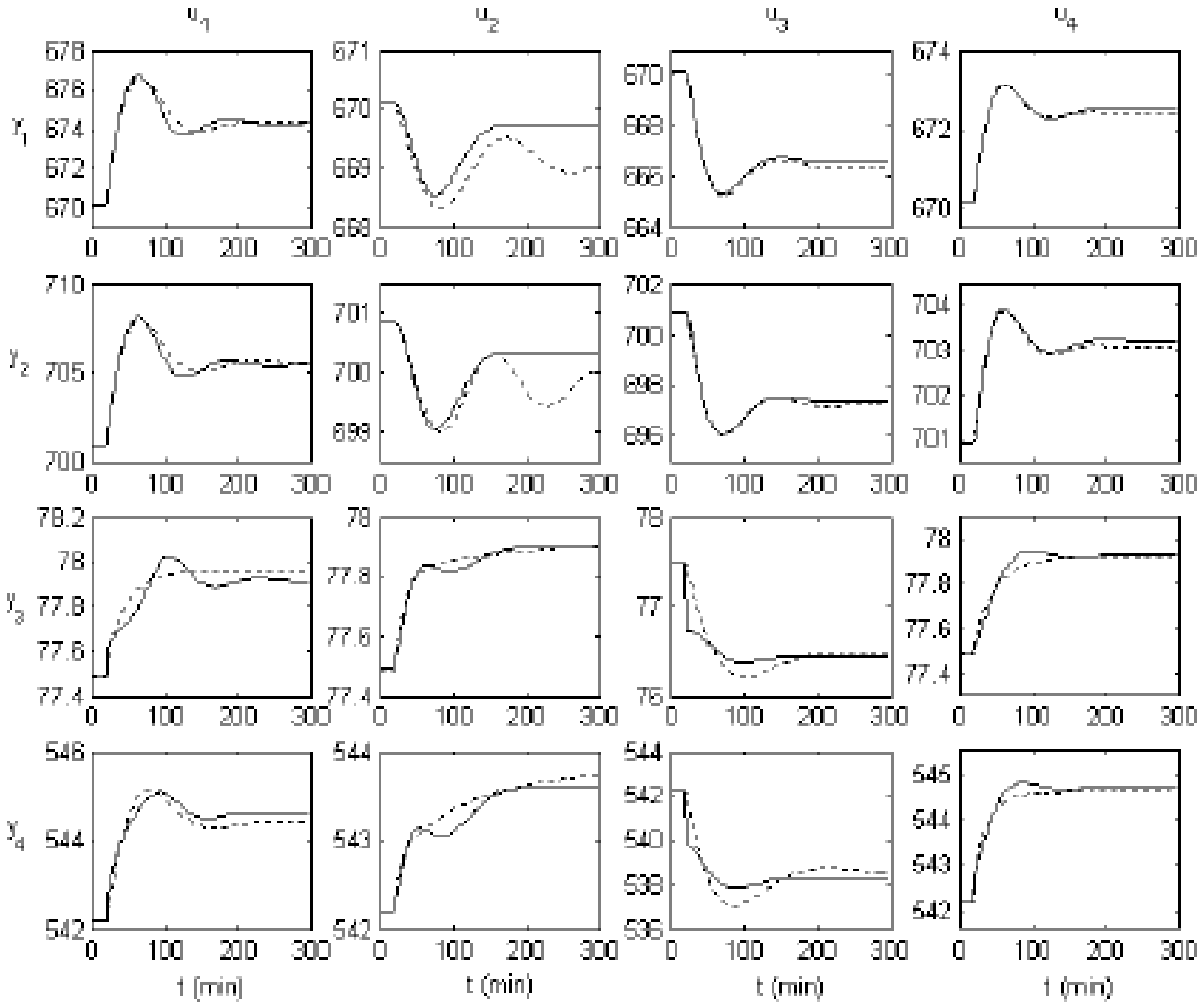
Linear systems - Step response

The step response is characterized by the following parameters:

- The *steady state value*, y_{ss}, of a step response is the final level of the output, assuming it converges.
- The *rise time*, T_r, is the amount of time required for the signal to go from 10% of its final value to 90% of its final value.
- The *overshoot*, M_p, is the percentage of the final value by which the signal initially rises above the final value.
- The *settling time*, T_s, is the amount of time required for the signal to stay within 5% of its final value for all future times.



Linear systems - Step response



Linearisation of non-linear models - 1

To linearise general non-linear systems, Taylor series expansion of a function is commonly used.

Recall that the Taylor series expansion of f(x) around a nominal point x_0 is given by

$$f(x) = f(x_0) + \left.\frac{df}{dx}\right|_{x=x_0} (x - x_0) + \frac{1}{2!}\left.\frac{d^2 f}{dx^2}\right|_{x=x_0} (x - x_0)^2 + \ldots \frac{1}{n!}\left.\frac{d^n f}{dx^n}\right|_{x=x_0} (x - x_0)^n + \ldots$$

Often this is written as

$$f(x) = f(x_0) + \left.\frac{df}{dx}\right|_{x=x_0} (x - x_0) + \quad \text{higher order terms}$$

For x_0 sufficiently close to x, the h.o.t are often omitted resulting in

$$f(x) \approx f(x_0) + a(x - x_0)$$

Linearisation of non-linear models - 2

Consider that x_0 is an equilibrium point of the system dx/dt= f(x), since we have dx/dt = 0 when x = x_0 (i.e., the system reaches a steady-state at x_0), we have the linear approximation of the dx/dt system as:

$$\dot{x} = \frac{dx}{dt} = f(x) \approx a(x - x_0)$$

To complete the linearisation, one defines a deviation state, z=x-x_0 where:

$$\dot{z} = az \quad where\ \dot{z} = \dot{x}$$

Note that this linear model is valid only in the neighbourhood of the nominal point x0, about which the linearisation is performed (i.e. term a is calculated).

Linearisation of non-linear models
Single state single input - 1

Consider a function f(x,u) which has equilibrium points at x0,u0 for the system dx/dt= f(x,u). The Taylor series expansion about the equilibrium point:

$$f(x,u) = f(x_0,u_0) + \left.\frac{df}{dx}\right|_{x,u=x_0,u_0}(x-x_0) + \left.\frac{1}{2!}\frac{d^2f}{dx^2}\right|_{x,u=x_0,u_0}(x-x_0)^2 + \ldots \left.\frac{1}{n!}\frac{d^nf}{dx^n}\right|_{x,u=x_0,u_0}(x-x_0)^n + \ldots$$

$$+ \left.\frac{df}{du}\right|_{x,u=x_0,u_0}(u-u_0) + \left.\frac{1}{2!}\frac{d^2f}{du^2}\right|_{x,u=x_0,u_0}(u-u_0)^2 + \ldots \left.\frac{1}{n!}\frac{d^nf}{du^n}\right|_{x,u=x_0,u_0}(u-u_0)^n + \ldots$$

Omitting h.o.t and defining deviation states, $z=x-x_0$ & $v=u-u_0$, the linear approximation of the nonlinear model becomes:

$$\dot{z} = az + bv$$

$$where \quad a = \left.\frac{df}{dx}\right|_{x,u=x_0,u_0} \qquad b = \left.\frac{df}{du}\right|_{x,u=x_0,u_0}$$

Linearisation of non-linear models
Single state single input - 2

Linearisation of algebraic nonlinear systems is done in similar fashion.

Consider a function y=g(x,u) which has equilibrium points at x0,u0 for the system, y_0= f(x_0,u_0). The Taylor series expansion around equilibrium point:

$$g(x,u) = g(x_0,u_0) + \left.\frac{dg}{dx}\right|_{x,u=x_0,u_0}(x-x_0) + \left.\frac{1}{2!}\frac{d^2g}{dx^2}\right|_{x,u=x_0,u_0}(x-x_0)^2 + \ldots \left.\frac{1}{n!}\frac{d^ng}{dx^n}\right|_{x,u=x_0,u_0}(x-x_0)^n + \ldots$$

$$+ \left.\frac{dg}{du}\right|_{x,u=x_0,u_0}(u-u_0) + \left.\frac{1}{2!}\frac{d^2g}{du^2}\right|_{x,u=x_0,u_0}(u-u_0)^2 + \ldots \left.\frac{1}{n!}\frac{d^ng}{du^n}\right|_{x,u=x_0,u_0}(u-u_0)^n + \ldots$$

Omitting h.o.t and using deviation states, z=x-x_0 & v=u-u_0 & w=y-y_0 the linear approximation of the algebraic nonlinear model becomes:

$$w = cz + dv$$

$$where \quad c = \left.\frac{dg}{dx}\right|_{x,u=x_0,u_0} \qquad d = \left.\frac{dg}{du}\right|_{x,u=x_0,u_0}$$

Linearisation of non-linear models Generalisation

Consider a general non-linear model with *n state variables, m input* variables, and *r output variables*

$$\dot{x}_1 = f_1(x_1, x_2, \ldots x_n, u_1, u_2, \ldots u_n)$$
$$\vdots$$
$$\dot{x}_n = f_n(x_1, x_2, \ldots x_n, u_1, u_2, \ldots u_n)$$
$$y_1 = g_1(x_1, x_2, \ldots x_n, u_1, u_2, \ldots u_n)$$
$$\vdots$$
$$y_r = g_n(x_1, x_2, \ldots x_n, u_1, u_2, \ldots u_n)$$

in vector notation

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$$
$$\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u})$$

First order Taylor series approximation of the non-linear model about the equilibrium point:

$z=x-x_0$ & $v=u-u_0$ & $w=y-y_0$

$$A_{ij} = \left.\frac{df_i}{dx_j}\right|_{x,u=x_0,u_0} \qquad B_{ik} = \left.\frac{df_i}{du_k}\right|_{x,u=x_0,u_0}$$

$$C_{lj} = \left.\frac{dg_l}{dx_j}\right|_{x,u=x_0,u_0} \qquad D_{lk} = \left.\frac{dg_l}{du_k}\right|_{x,u=x_0,u_0}$$

in state – space form

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{v}$$
$$\mathbf{w} = \mathbf{C}\mathbf{z} + \mathbf{D}\mathbf{v}$$

with matrice dimensions : **Anxn, Bnxm, Crxn, Drxm**

Example 1

Linearise the following non-linear model about the equilibrium point: $x_1=1$, $x_2=0$, $u=0$

$$\dot{x}_1 = x_1^2 + \sin x_2 - 1$$
$$\dot{x}_2 = -x_2^3 + u$$
$$y = 2x_1 + x_2$$

Write the linear model in state-space form.

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}, g = [g], \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, u = [u], y = [y]$$

$$A = \begin{bmatrix} a_{11} = \left.\frac{df_1}{dx_1}\right| & a_{12} = \left.\frac{df_1}{dx_2}\right| \\ a_{21} = \left.\frac{df_2}{dx_1}\right| & a_{22} = \left.\frac{df_2}{dx_2}\right| \end{bmatrix} = \begin{bmatrix} 2x_1 & cos(x_2) \\ 0 & -3{x_2}^2 \end{bmatrix}_{(x_1,x_2,u=1,0,0)} = \begin{bmatrix} 2 & 1 \\ 0 & 0 \end{bmatrix}$$

Using $\mathbf{z} = \begin{bmatrix} x_1 - 1 \\ x_2 \end{bmatrix}, \mathbf{v} = [u]$ *and* $w = y - 2$

The state-space model

$$\dot{\mathbf{z}} = \begin{bmatrix} 2 & 1 \\ 0 & 0 \end{bmatrix}\mathbf{z} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}\mathbf{v}$$

$$\mathbf{w} = \begin{bmatrix} 2 & 1 \end{bmatrix}\mathbf{z}$$

$$B = \begin{bmatrix} b_{11} = \left.\frac{df_1}{du}\right| \\ b_{21} = \left.\frac{df_2}{du}\right| \end{bmatrix}_{(x_1,x_2,u=1,0,0)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} c_{11} = \left.\frac{dg}{dx_1}\right| & c_{21} = \left.\frac{dg}{dx_2}\right| \end{bmatrix}_{(x_1,x_2,u=1,0,0)} = \begin{bmatrix} 2 & 1 \end{bmatrix}$$

Linearisation using symbolic computation in matlab

```
syms x1 x2 u
f1=x1^2+sin(x2)-1;
f2=-x2^3+u;
g1=2*x1+x2;

As=jacobian([f1;f2],[x1;x2]);
Bs=jacobian([f1;f2],[u]);
Cs=jacobian([g1],[x1;x2]);
Ds=jacobian([g1],[u]);

% linearisation point (e.g.
equilibrium point)
x1=1; x2=0; u=0;
A=eval(As);
B=eval(Bs);
C=eval(Cs);
D=eval(Ds);
```

Uses symbolic language

The matlab function jacobian Provides the linear approximation

The matlab function ”eval” useful to evaluate the jacobian at the linearisation point

Linearisation using Matlab – Simulink

Example 2: Linearisation of predatory prey model

$$\frac{dH}{dt} = rH\left(1 - \frac{H}{H_{\max}}\right) - \frac{aHL}{c + aHT_h}; \quad H \geq 0$$

$$\frac{dL}{dt} = bL\left(1 - \frac{L}{kH}\right); \quad L \geq 0$$

Use matlab/simulink to linearise the non-linear systems:
Step 1. Implement the non-linear system as matlab m-file s-fun
Step 2. Describe the system as a Simulink block
Step 3. Linearise the system using *trim* and *linmod*

Matlab tools for linearisation of models

Step 1 m-file s-fun

```
function [sys,x0,str,ts] =
predprey_sfun(t,x,u,flag,xinit,par)

dxdt(1)= (r)*x(1)*(1-x(1)/Hmax) - a*x(1)*x(2) / (c+ x(1));
dxdt(2)= b* a*x(1)*x(2) / (c+x(1)) - d*x(2);
sys = [dxdt(1) dxdt(2)];
```

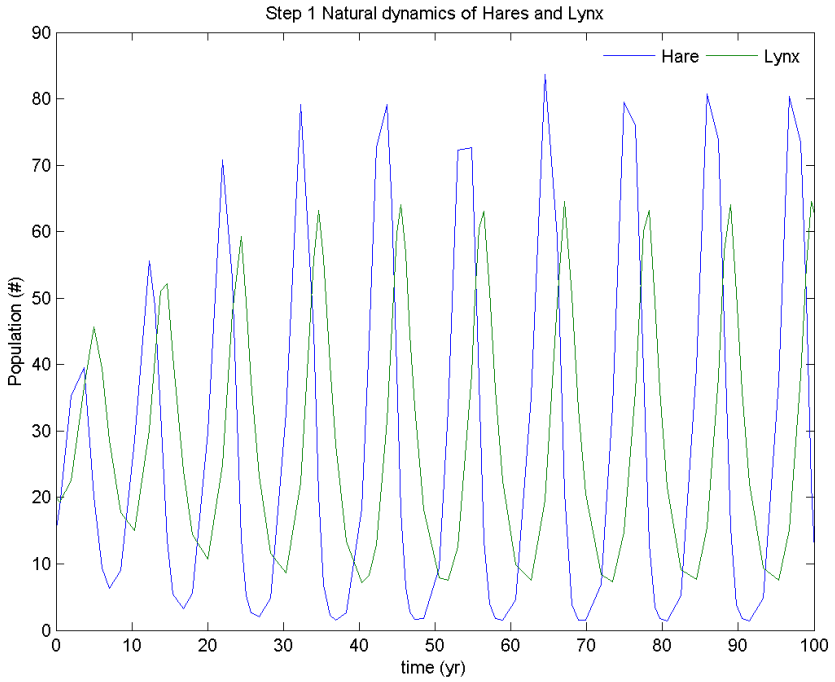
$$\frac{dH}{dt} = rH\left(1-\frac{H}{H_{\max}}\right)-\frac{aHL}{c+aHT_h}; \quad H \geq 0$$

$$\frac{dL}{dt} = bL\left(1-\frac{L}{kH}\right); \quad L \geq 0$$

Solve and see the input – output dynamics

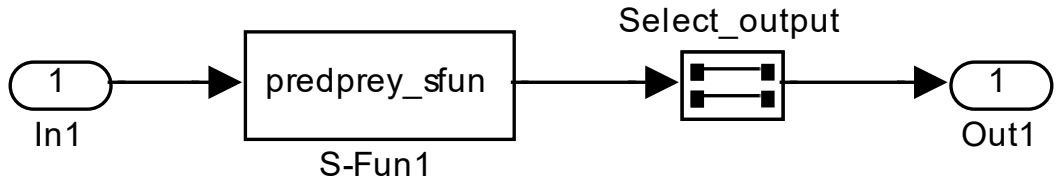
```
% initial conditions
xinit = [15 20]; u0 = r;

%% simulate the system
sim('predprey_nonlin',time)
```



Matlab tools for linearisation of models

Step 2 simulink block



Step 3 linearise the model

```
%% provide the nominal point
(for control apps, the steady-sate point
[xe ue] = trim('predprey_nonlin');
ue=1.6;   % user defined value for the input

%% linearize the system around trim point
[A,B,C,D] = linmod('predprey_port', xe, ue);
```

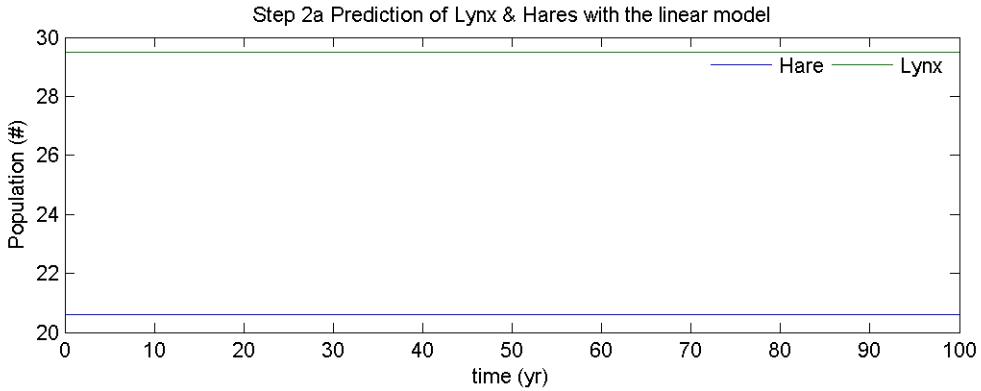
$$A = \begin{bmatrix} 0.1263 & -0.9333 \\ 0.5680 & -0.0000 \end{bmatrix} \quad B = \begin{bmatrix} 17.1972 \\ 0 \end{bmatrix}$$

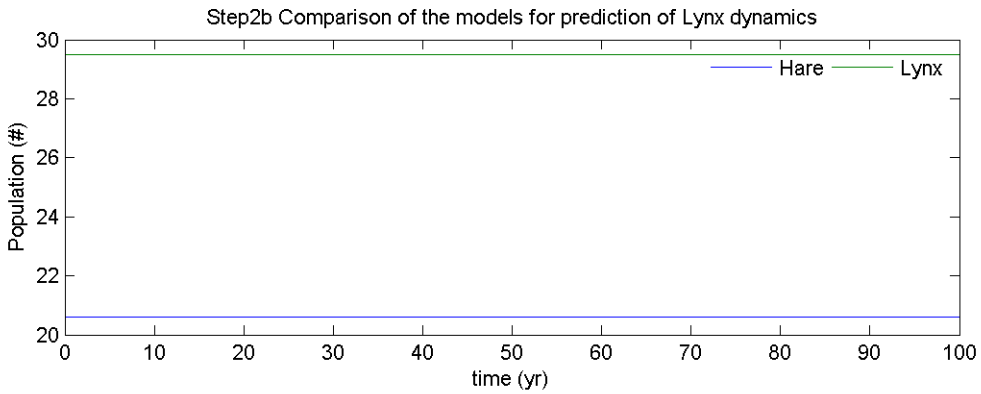
$$C = \begin{bmatrix} 1.0000 & 0 \\ 0 & 1.0000 \end{bmatrix} \quad D = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Matlab tools for linearisation of models

Step 4 compare the models with each other

```
u=ue; x0=xe;
lsys = ss(A,B,C,D); dt = 1;
tl = t0:dt:tfin; % time should be
equidistant
v = (u-ue) * ones([length(tl),1])
; % v is the deviation variable:
v=u-ue ; v must have the same
numbers as time vector
z0 = (x0-xe); % the deviation
variable
[wl,tl,zl]= lsim(lsys,v,tl,z0) ;
xev = xe * ones([length(tl),1])';
% xe (a scalar) converted to a
vector of values
xl = zl + xev' ;
yl = wl + xev';
```

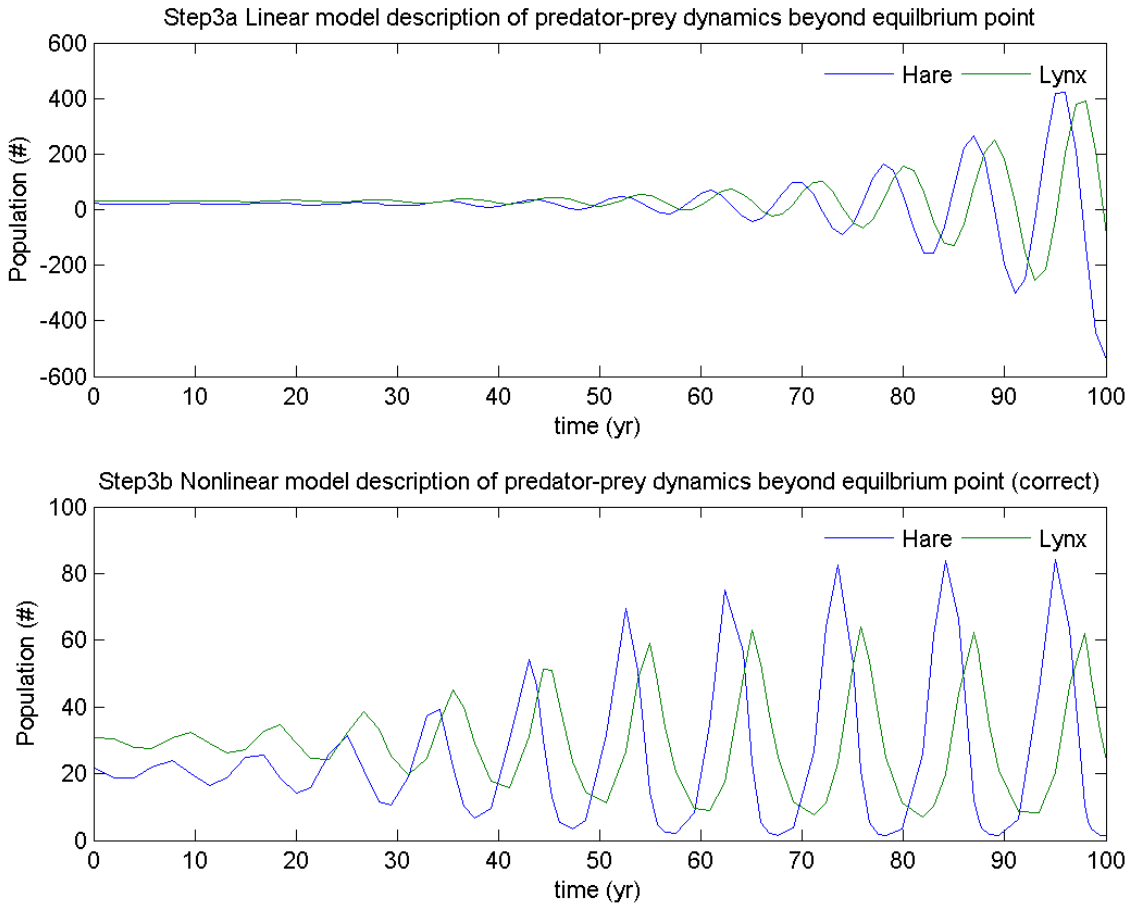




Matlab tools for linearisation of models

Step 4 compare the models with each other

The linear model is valid only in a small neighbour-hood around the point of linearisation



State-space model to transfer function

Consider a general state-space model with *n state variables, m input* variables, and *r output variables*

$$\frac{dx}{dt} = Ax + Bu \quad x \in R^n, u \in R$$
$$y = Cx \quad y \in R$$

Laplace Transform of the state-space model using properties of Laplace (e.g. differentiation (df/dt = sF(s) – f(0))):

$$sX(s) - x(0) = AX(s) + BU(s)$$
$$Y(s) = CX(s)$$

Rearranging and solving for X(s) yields:

$$X(s) = (sI - A)^{-1}x(0) + (sI - A)^{-1}BU(s)$$

Transfer function, H(s), of the above state-space model at x(0)=0, is then:

$$\frac{Y(s)}{U(s)} = H(s) = C(sI - A)^{-1}B$$

Note that here H(s) is derived for a single input and single output system. It can be generalised for multiple inputs and outputs.

Example 3

Calculate the transfer function for the state space model

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 4 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x$$

```
% describe state-space model
A=[0 1;
-2 -3]; B=[0 4];
C=[1 0];
H=C*(s*eye(2)-A)^-1*B;
pretty(H)
```

Solution

$$H(s) = \frac{4}{s^2 + 3s + 2}$$

Is the system stable?

Answer: check eigs(A) as well as roots of denominator of H(s)?

Eigs(A)=[-2,-1] and root of denominator = (-2,-1)

Local Stability of non-linear Systems

By studying the properties of A:

Linearization about equilibrium point captures “tangent” dynamics, hence:

- If the linearised system is asymptotically stable, then it implies *local asymptotic stability* of equilibrium point ($\lambda(A) < 0$))
- If linearisation is *stable but not asymptotically stable, nothing can be conclude* about the non-linear system (if $\lambda(A) = 0$ but not $\lambda(A) < 0$)
- If linearisation is *unstable, it can be concluded that nonlinear system is locally* unstable ($\lambda(A) > 0$))

Local approximation particularly appropriate for control systems design

- Control is often used to *ensure that a system stays near a desired equilibrium point*
- If dynamics are well-approximated by linearisation near this equilibrium point, you can use this to design the controller that keeps the system there (!)

Wrap-up: linear systems

- Linearity with respect to initial condition and inputs.
- Stability characterized by eigenvalues of **A**.
- Many applications and tools available for linear systems.
- Provide local description for nonlinear systems.
  – No use of the linearised system other than at the point of initialisation!
- Non-linear systems can be linearised using Taylor series expansion of functions or by system identification (simulation or experiments).
- Symbolic (analytical) as well as numerical (simulink) tools are available to linearise non-linear models.

Linearisation vs. step test identification

In case we have a detailed first principle model of a process we can either linearise it or simulate a step test and identify the linear model.

The advantage of a scheme where we can fast-compute a relevant linear model by linearisation is that we can easily update our model approximation if the point of operation changes (or the nonlinear model itself).

The advantage of the simulated step test is that we only include observed dynamics and we are ensured a low order approximation, independet of the state dimension of the non-linear system. It may therefore be the preferred approach for very larger state space models.

Exercise

Solve exercise 4A : Obtain a linear model of the non-linear fermenter-system!

See the details in the exercise 4 description...

Hint: You may use either Matlab Symbolic Math tools or the Matlab-Simulink implementation. Both should help you to get the job done (the latter is suggested).