

## CHAPTER 9: ADAPTIVE CONTROL

### 9.1. Adaptive control for industrial use1

Awarded control technology .....	1
Industrial applications.....	1
Adaptive control and PID.....	2

### 9.2. What is adaptive control? ..... 3

Different types of adaptive controllers.....	3
A general self-adaptive controller .....	4
The adaptive control method.....	5
The adapting part. ....	5
Regulating part. ....	6
Advanced course.....	8

### 9.3. STEGX2- the adaptive regulator....11

Default settings of parameters .....	11
Model feedback (NA, NB ,MD) .....	12
Default values .....	12
Number of controller parameters.....	12
Process time delays.....	12
Model feedforward (NC,ALFA) .....	13
Default values .....	13
Design considerations.....	13
Measurements (MV, FF1, FF2) .....	15
Conditioning the measurement .....	15

Noise filter. ....	15
Regulator filtering. ....	15
Setpoint (SP) .....	16
Man/auto switching (AUTO, ADAPT, LOAD) 16	
Controller response (SAMP, POLE) .....	17
Default values .....	17
The feedback response.....	17
The feedforward response. ....	18
Non-minimum phase processes .....	20
Protective network (RESU, RESY, UMAX, UMIN).....	20
Control limits (HI, LO,DUP,DUM).....	21
Mode switching (BMPLVL) .....	21
Adaption window (W) .....	22
Start-up procedure for STEGX2 module....	23
Preparations:.....	23
Adjustments .....	23
Afterwards: .....	23



## 9.1. Adaptive control for industrial use

### Awarded control technology

First Control has used adaptive control for many years. Installations have been made in many different types of industrial processes, ranging from energy plants and chemical plants to steel mills and paper mills. First Control's adaptive regulators are well-proven and efficient controllers. Experience from field studies shows that a 50-75% improvement in control accuracy can be expected using the adaptive regulator STREGX2 instead of PID controllers.

A high-performance control system must also include all the logics, filtering and computations the needed for a good control design. The MicroController contains an extensive function library with more than 150 different function modules which gives the user all the freedom he needs to do the computations and signal handling he wants. For more details see chapter 8.

The inventor of the adaptive control algorithms used by First Control received the *IEEE CSS Control Technology Award* from the American Engineering organization IEEE for his contributions to the adaptive control technology for industrial use.

### Industrial applications.

Some of the processes where adaptive regulators have been used by First Control are listed below. The adaptive regulators used in the installations are all included in the function library of the MicroController XC01 control system.

- Continuous casters
- Cold rolling mills
- Steel furnaces
- Annealing lines
- Hot rolling mills
- Paper machines
- District heating plant
- Steam boilers
- Power plants
- Gas turbines
- Chemical plants

In case you are interested in specific solutions for these or similar industrial processes, please contact First Control for more information.

The adaptive regulator STREGX2 is designed for any type of control and is easy to use without knowing the theoretical details. Some background information is provided below.

## Adaptive control and PID.

The STREGX2 adaptive regulator is used in about the same way as a conventional PID, i.e. the purpose is to keep a measured value as close as possible to a given set point. The adaptive regulator has, however, a much more advanced internal structure and it is capable to change its own behaviour if the process changes. Therefore, the STREGX2 regulator is considerably more accurate in controlling an industrial process.

The improvements obtained with the adaptive controller STREGX2 compared to conventional PID controllers or more restricted adaptive controllers are mainly due to the following:

- ❑ Many more regulator parameters can be used
- ❑ The controller adapts to the process at start up
- ❑ The commissioning time is shortened
- ❑ The controller adapts to changes in the process dynamics
- ❑ Adaptive feedforward control can be used in an efficient way
- ❑ The control action is based on predicted future process behaviour
- ❑ No manual tuning of regulator parameters is necessary
- ❑ The regulator may operate at a much larger production range
- ❑ In case instabilities occur, they will normally soon disappear.
- ❑ The control accuracy is normally improved by 50-75%

The advanced features of the controller also have the advantage that the user can focus on the control design rather than on time-consuming manual tuning.

## 9.2. What is adaptive control?

The notion of *adaptive control* has been used to categorize quite different types of regulators ranging from simple PID controllers with programmable gains to true self-adaptive controllers. Many of the controllers referred as “adaptive” are in fact not adaptive in the true sense. For this reason, we will explain the notion of adaptive control in a little more details.

### Different types of adaptive controllers.

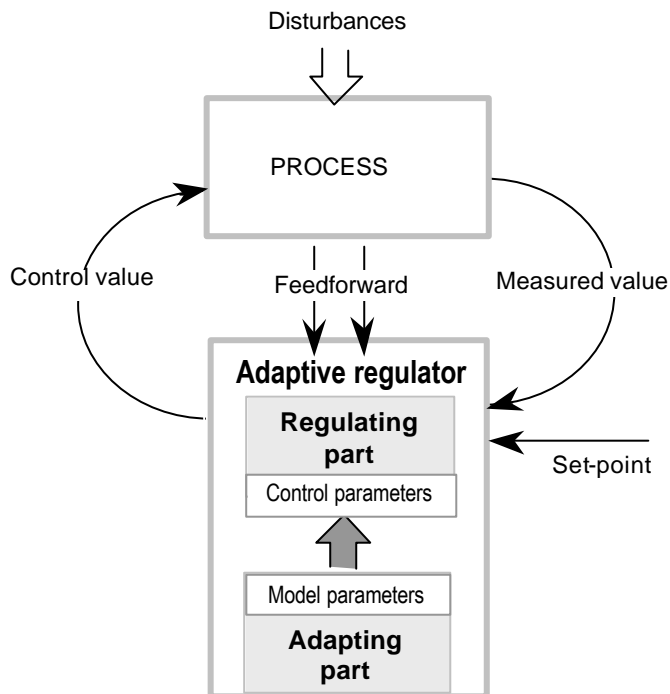
A truly *adaptive controller* is capable of learning from previous events to improve future performance. This can be achieved in many different ways, but a common feature of all adaptive controllers is that they have a much “longer” memory than e.g. a normal PID regulator. The “long” memory can e.g. be a parametric process model where the model parameters are identified by means of some recursive identification method.

There are basically four different types of regulators which are often referred as adaptive:

<b>PID with Gain Scheduling</b>	The control parameter are changed during running as a predefined function of process measurements. Sometimes the regulator uses several sets of gains, selected using the process status. In modern control design, similar compensating features are sometimes added to truly adaptive controllers to compensate for known non-linearities or known gain changes.	Not truly adaptive
<b>Autotuner</b>	An autotuner is often a PID controller where the control parameters are automatically tuned only at commissioning. Often the regulator applies a special procedure at start-up, e.g. a sequence of step responses or a relay type of control, in order to find an appropriate parameter setting. After commissioning, the parameters are fixed until the user starts the tuning procedure all over again.	Not truly adaptive
<b>Adaptive PID</b>	Many true adaptive controllers are of PID type. The small number of parameters makes the construction easier for the supplier, but many of the advantages with adaptive control can not be used since the controller structure is too simple. For instance, controlling a process with time-delay requires more internal structure. Special-designed adaptive PID controllers are also often used with special objects, e.g. motor drives.	Truly adaptive

<b>General adaptive regulators</b>	The general-purpose adaptive regulators are designed to control more or less any type of process. There is no real limitation of the number control parameters other than maybe from a practical point of view. The STREGX2 regulator used by First control may have as many as 20 regulator parameters adapted automatically . It includes adaptive feedforward control and adaptive time delay compensation inherent in the control structure.	Truly adaptive
------------------------------------	--	----------------

### A general self-adaptive controller



An adaptive regulator like the STREGX2 controller in the MicroController function library consists internally of two parts:

- **A regulating part** which performs all the control. The regulator is similar to an ordinary PID-regulator, but contains normally many more control parameters. In addition to closed-loop control, the regulating part can also do feedforward control and time delay compensation.
- **An adapting part** which creates a model of the process and computes new values on all the control parameters when the model changes. The adapting process can be controlled by inputs on the regulator module, but is normally active all the time .

Due to the rather heavy computations, the adapting part of the MicroController software is run on a special *background priority level*, located below the priority levels for the application programs. In practise, this means that the adaptation is performed during CPU spare time, i.e. as fast as possible without disturbing the normal control functions. It also means that the self-adaptive regulator can be run with fast sampling since there is seldom a need for adapting the control parameters at each sampling instance of the regulator.

Unlike most other adaptive regulators on the market, First Control's adaptive regulator STREGX2 perform the adaptation by itself and with no more information other than the measured value, the control value and the feedforward signals. The natural variations in the process will be sufficient for the regulator to create its process model.

The adaption process must be performed in a slower time-scale than the control. This means that the speed by which the adaptive controller can change its regulator parameters is limited. The user can specify the time-scale by a certain parameter in the STREGX2 module, *the adaptation window (W)*. Normally, it takes *W* regulator samples for the controller to completely retune itself after a large change in the process dynamics (e.g. a sudden gain change).

## The adaptive control method

The description below applies to the STREGX2 self-adaptive regulator in the MicroController function library, described in chapter 8.

### The adapting part.

The adaptive regulator is internally based on a sampled (time-discrete) process model describing the process behaviour at the sampling instances:

$$mv(t) = \sum_{i=1}^{NA} a(i) \cdot mv(t-i) + \sum_{i=1}^{NB} b(i) \cdot u(t-MD-i) + \sum_{i=1}^{NC} c1(i) \cdot ff1(t-MD-i) + \sum_{i=1}^{NC} c2(i) \cdot ff2(t-MD-i)$$

where  $t, t-1, t-2, \dots$  etc are the sampling instances and  $t$  is the present time. Furthermore:

$mv$	= feedback measured value MV
$u$	= control value U
$ff1$	= feedforward measured value FF1
$ff2$	= feedforward measured value FF2
$a(i), b(i), c1(i), c2(i)$	= model parameters
$MD$	= a lower limit on the time delay in the process
$NA$	= number of parameters $a(i)$
$NB$	= number of parameters $b(i)$
$NC$	= number of parameters $c1(i)$ and $c2(i)$ (always the same number)

The model is a general linear dynamic system whose order is determined by the parameters  $NA, NB, NC$  and  $MD$ . The model gives a good description of the performance of the process close to

the current operating point. Basically, the model describes the predicted future behaviour of the measurement value  $mv(t)$  the next few samples.

The STREGX2 controller uses an *recursive extended least-square identification* algorithm to compute the model parameters  $a(i)$ ,  $b(i)$ ,  $c1(i)$  and  $c2(i)$  from measurement data. The model parameters will be changed slightly each time a new set of measurement data is read. No deliberate process disturbances or test signals are needed to make the identification work.

One major difficulty in the construction of adaptive controllers for practical use is the following. The better the adaptive controller works, the less variation there is in the measurement  $mv(t)$ . Since the identification algorithm is based on natural variations only, the accuracy of the model will then degrade when the control accuracy improves unless protective actions are taken. It is absolutely crucial that a *protective network* is added to the identification algorithm which prevents the model from degrading when the information contents in the measurements is low.

In the STREGX2 case, the protective network is integrated in the module software. The protective software is based on more than 10 years of field experience using adaptive controllers in industrial processes. It is also possible to restrict the controller adaption in the application program, by switching the adaption on/off or load a new parameter set.

To be adaptive, the controller must also be capable to change its process model if the process changes. The user defines the allowed rate of change by specifying *the adaption window (W)* in the STREGX2 module. The adaption window can be interpreted as the amount of old data used by the controller to form its present process model.

#### Regulating part.

The *regulating part* uses a *pole placement* method to compute the controller parameters to be used in the control. The computation is based on the identified process model. In the STREGX2 case, the regulator is designed to always contain an integrating part to remove control off-sets. The control strategy will be of the form:

$$u(t) = u(t-1) + \sum_{i=0}^{NG} g(i) \cdot mv(t-i) + \sum_{i=1}^{NF} f(i) \cdot u(t-MD-i) + \sum_{i=1}^{NH} h1(i) \cdot ff1(t-MD-i) + \sum_{i=1}^{NH} h2(i) \cdot ff2(t-MD-i)$$

i.e. the control value  $u(t)$  is computed as a weighted sum of the most recent measurement values, control values and feedforward values. The coefficients  $g(i)$ ,  $f(i)$ ,  $h1(i)$  and  $h2(i)$  are the controller parameters.

Note that the PI controller is a special case with  $NG=2$ ,  $NF=0$ ,  $NH=0$  in the formula above. One could say that the PI controller is "optimal" for a very low order system model. This is also the reason why PI controllers have difficulties in controlling processes with time delays or more complex dynamics. The STREGX2 controller does not have the same difficulty, since it works with a more complex internal structure and many more control parameters.

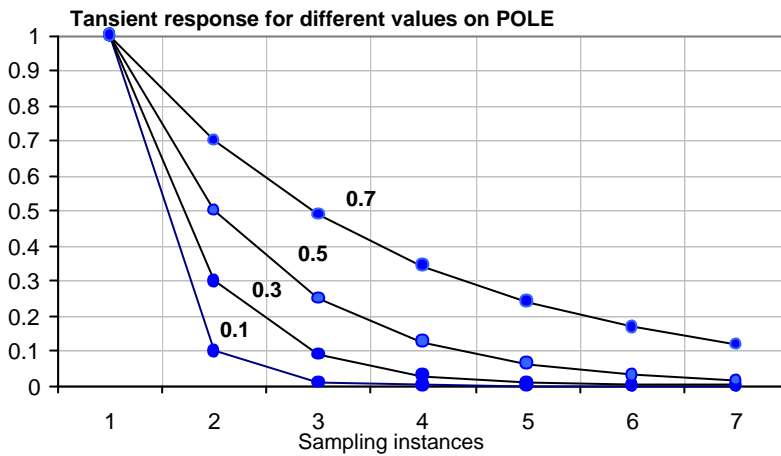


The user may shape the controller response by specifying one dominant pole position (POLE) . The remaining poles are placed at the origin (0). The POLE positions determines in principal the transient behaviour of the closed-loop response after a disturbance as

$$e(t) = \text{POLE} * e(t-1)$$

where  $e(.)$  is the control error and  $t, t-1, \dots$  are the sampling instances.

The transient responses for different pole values are shown in the figure:



## Advanced course

This is for you who are somewhat more familiar with adaptive control technology. If you find it too difficult, proceed to the next chapter. It is not necessary to understand the contents of this chapter to be able to use the adaptive regulator.

The *model* used by the adaptive regulator is of the form

$$A(z) \cdot mv(t) = z^{MD} (B(z) \cdot u(t) + C1(z) \cdot ff1(t) + C2(z) \cdot ff2(t))$$

where  $z$  is the backward shift operator and  $A(z)$ ,  $B(z)$ ,  $C1(z)$  and  $C2(z)$  are polynomials in  $z$  of degrees  $NA$ ,  $NB-1$ ,  $NC-1$  and  $NC-1$  respectively. The minimal delay is  $MD$ . The model is thus a general time-discrete linear dynamic system. The number of model parameters that may be used is only restricted by the size of the internal database and the computational power. As a guidance, it is possible to select as many as 20 model parameters in a STREGX2 regulator. If you exceed the limit, the MicroController system supervisor will urge you to decrease the number of parameters before you are allowed to proceed.

The *adapting part* uses a "modified" recursive least-square algorithm to identify all the parameters in the polynomials  $A(z)$ ,  $B(z)$ ,  $C1(z)$  and  $C2(z)$ . The algorithm uses a forgetting factor  $\lambda$  computed as

$$\lambda = 1 - 1/W \quad ; \quad W = \text{adaption window in STREGX2 parameters.}$$

The reason to use the notion of "Adaption window" instead of the forgetting factor is that it is easier to relate to the specific process at hand.

The standard recursive least-square algorithm is modified with a *protective network* to be able to cope with:

- ❑ Large model errors (residuals) that may wipe out the present process model.
- ❑ Small variations in data that may cause the process model to degrade.
- ❑ Special disturbances causing “non-exciting” situations.
- ❑ Model data outside realistic limits.
- ❑ Computational sensitivity in the variance matrix.

The special software that implements the protective network is based on extensive field experience from many installations in industrial processes.

The *feedback regulator* is computed using a *pole placement* method, i.e. by finding the minimal order solution  $F(z)$ ,  $G(z)$  to the diophantic equation:

$$A'(z)(1-z) \cdot F(z) + z^M D B'(z) \cdot G(z) = 1 - \text{POLE} \cdot z$$

where  $A'(z)$  and  $B'(z)$  corresponds to  $A(z)$  and  $B(z)$  with the common factors removed. In practice, the identified polynomials  $A(z)$  and  $B(z)$  will nearly always contain "near" common factors which model the (uncontrollable) disturbances acting on the system. The "near" common factors will result in excess regulator gains and bad control if not removed. Therefore, First Control has developed an efficient algorithm which finds the unique minimal order solution at the same time as the "near" common factors are cancelled.

The *feedback control value*  $u_{FB}(t)$  then becomes

$$(1-z)F(z) \cdot u_{FB}(t) = -(G(z) \cdot mv(t) - Hr(z) \cdot sp(t))$$

where the polynomial  $Hr(z)$  has the same steady-state gain as  $G(z)$ , i.e.  $Hr(0)=G(0)$ . The factor  $1-z$  will ensure that the regulator has integral action. The polynomial  $Hr(z)$  will ensure that the responses for changes in the set-point are well-behaved.

The *feedforward regulator* attempts to eliminate the disturbances before they influence the controlled variables. A feedforward regulator that makes exact feedforward control in the model above is  $B(z) \cdot u(t) = -C1 \cdot ff1(t) - C2(z) \cdot ff2(t)$ . If the process is non-minimum phase, i.e.  $B(z)$  is unstable, exact feedforward control will result in an unstable system. We then have to look for approximative stable solutions. In order to ensure stability, the user can specify the desired degree of stability with the parameter ALFA (all poles will be within a circle of radius ALFA/20).

The *feedforward control value*  $u_{FF}(t)$  then becomes

$$B'(z) \cdot u_{FF}(t) = -C1'(z) \cdot ff1(t) - C2'(z) \cdot ff2(t)$$

where  $B'(z)$ ,  $C1'(z)$  and  $C2'(z)$  are approximations of  $B(z)$ ,  $C1(z)$  and  $C2(z)$  respectively. The approximation is performed so that  $B'(z)$  has all its poles within a circle of radius  $ALFA/20$ . Stability is ensured if  $ALFA \neq 0$ . If  $ALFA$  is chosen large enough, then  $B'(z)$ ,  $C1'(z)$  and  $C2'(z)$  becomes equal to  $B(z)$ ,  $C1(z)$  and  $C2(z)$  respectively, and we have the case of exact feedforward. If  $ALFA=0$ , the feedforward path will contain only poles at the origin (0).

The *combined feedback-feedforward* regulator is then composed as the sum of the *feedback* path and the *feedforward* path

$$u(t) = u_{FB}(t) + u_{FF}(t)$$

Since the feedback and feedforward paths are computed separately, they can also be separated in the control design. In the STREGX2 module, the user selects the combination he wants by the MODE parameter. The available control modes are:

0. Feedback control only
1. Feedback control and one feedforward link.
2. Feedback control and two feedforward links.
3. Feedforward control with one link only.
4. Feedforward control with two links.

## 9.3. STEGX2 - the adaptive regulator.

### Default settings of parameters

STREGX2	
- L1	ON
- L2	AUTO
- L3	ADAPT
- L4	LOAD
- L5	DUMP
- I1	MODEL
- R1	MV
- R2	SP
- R3	UE
- R4	FF1
- R5	FF2
- R6	HI
- R7	LO
- R8	DUP
- R9	DUM
I1	NA
I2	NB
I3	NC
I4	MD
I5	ALFA
I6	SAMP
R1	UMAX
R2	UMIN
R3	RESU
R4	RESY
R5	KINIT
R6	BMPLVL
R7	POLE
I7	MODE
I8	W

The function module STREGX2 in the MicroController function library is shown to the left. A brief outline of the methods used in the regulator software is explained in chapter 9.2.

The parametrization of the STREGX2 regulator is normally done very much by *default settings* which will work in almost all cases. Once an adaptive regulator is defined in your application, you can reuse the settings by copying your first design and then make the changes you want.

The default settings of regulator parameters are as follows:

NA	=	3
NB	=	5
NC	=	3
MD	=	1
ALFA	=	20
SAMP	=	Select 30-60% of desired response time.
UMAX	=	Max value in control range
UMIN	=	Min value in control range
RESU	=	0.05% of control range UMAX-UMIN
RESY	=	0.05% of measurement range
KINIT	=	0.1
BMPLVL	=	Allowed change in control value at mode change
POLE	=	0.5
MODE	=	Select the control mode: 0=FB only, 1= FB+FF1, 2=FB+FF1+FF2, 3= FF1 only, 4=FF1+FF2
W	=	100

This setting will work in most cases. In order to make the control faster, select a smaller value on the SAMP parameter value. In order to make the control slower, increase the SAMP parameter value.

## Model feedback (NA, NB ,MD)

### Default values

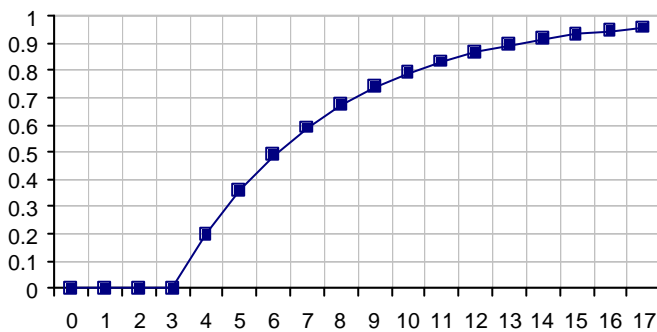
NA	3
NB	5
MD	1

### Number of controller parameters.

The number of model parameters the regulator uses internally to calculate the feedback control law is determined by the integer parameters NA, NB and MD. The number of controller parameters becomes normally  $NA + NB + MD - 1$ . The regulator will by itself reduce the complexity if the model is too large, i.e. there is no real drawback in using too many parameters other than that the amount of computations will increase especially in the adapting part.

### Process time delays

Sometimes there is a considerable time delay in the process, i.e. the process will not response at all on a control actions until a certain time has elapsed. Consider the step response in the figure where 0 is the point of time where the step was made.



The squares on the curve denotes the values at the regulator sample instances as is defined by the parameter SAMP in the STREGX2 module. A process time delay sets an absolute limit how fast the feedback control can be. In addition, the time delay will heavily affect the closed-loop stability. A well tuned PID controller can normal control a step disturbances within 3-5 times the time delay with enough stability margins. An adaptive controller can normally do the same thing within 1.5 - 2 times the time delay.

The *sampled time delay (D)* is the number of regulator samplings until the process responds. In the example above, the sampled time delay  $D=4$ , since the process responds for the first time after 4 sampling instances. Note that the sampled time delay is affected by the SAMP parameter (see below). If the regulator sampling time is doubled in the figure, the sampled time-delay becomes  $D= 2$  instead.

Then the following conditions on MD and NB must be satisfied:

- MD must be less than or equal to the sampled time delay D
- MD + NB must be larger than the sampled time delay D

For instance, MD=2 and NB=8 is an allowed selection of parameter values in the example above. In this case, the process delay may vary between 2 and 8 regulator samples.

#### Rules of thumb

- Select the SAMP parameter so that you will have 1-3 sampling instances on the time delay.
- If the time delay does not vary, select the parameters MD equal to the sampled time delay and NB according to default.
- If the time delay varies, select MD equal to the smallest time delay in the operation range and select NB equal to the largest time delay in the operation range + 3 - MD.

Remark. Changes in time delays are handled by the regulator. However, a sudden change in the time delay is a severe change in process dynamics from a control point of view. It may very well happen that the control will be severely degraded until the controller has had enough time to retune itself. If possible, design your control system so that the time delay is fixed. For instance, you can select a sampling which is proportional to the transportation speed in case you have a transportation delay.

## Model feedforward (NC,ALFA)

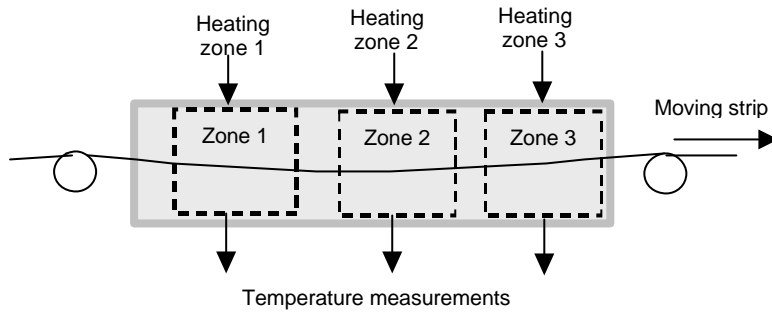
### Default values

NC	3
ALFA	20

### Design considerations

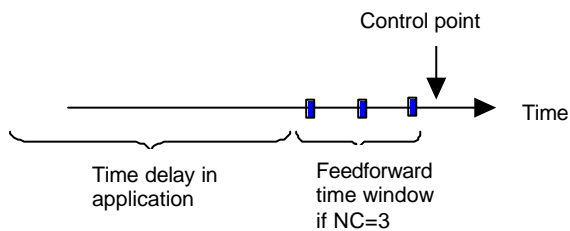
Feedforward control means that a process disturbance is compensated before it upsets the process. To be able to do this, the regulator needs advance information about the disturbance. A feedforward measurement provides such information. Since many processes are sequential, i.e. the different production steps follows one after the other, there is often large possibilities to use feedforward control. The adaptive feedforward control is the one property of the STREGX2 regulator that improve the process control most.

Consider the following simple example showing a moving steel strip in a furnace:



If a temperature disturbance is measured in zone 1, the same disturbance will affect zone 2 some time later. The temperature measurement of zone 1 then provide *advance information* about a disturbance that soon will enter zone 2, and can therefore be used for feedforward control in zone 2.

In the example above, it is important that the controlled heating of zone 2 is changed in exactly the right moment when the disturbance enters zone 2. The NC parameter provides a *time window* for the feedforward control within which the STREGX2 regulator may adjust the control action for a feedforward disturbance:



The feedforward signals should be presented at the FF1 or FF2 inputs prior to the point of time when the control action should be taken. If needed, the signals must be delayed up to the time window by the MDEL2 module, letting the regulator make the fine adjustment.

By the parameter ALFA you achieve a specified degree of stability in the feedforward control. If you choose ALFA=20 you are guaranteed a stable feedforward control, see section 2 "Advanced course" for more information about feedforward stability.

#### Rules of thumb

- ❑ Select the SAMP parameter so that the control will be accurate enough in time.
- ❑ Select the default value for the NC parameter, i.e. NC=3.
- ❑ Select the default value for the ALFA parameter, i.e. ALFA=20.
- ❑ Make a delay in your application programs so that the feedforward signals are presented at the FF1 and FF2 inputs approximatively NC sampling times before the control action should take place.



## Measurements (MV, FF1, FF2)

### Conditioning the measurement

The measurements should always be conditioned before used in any type of control. Normally the conditioning is done by filters on two levels

1. *Noise filtering*, removing high-frequency noise
2. *Regulator filtering*, improving regulator information

Be very careful of how the measurements are filtered. A heavily filtered signal may look nice to the viewer, but it is very bad for control. A signal filter always produces a negative phase shift in the system, making the control less stable. Check also the filter times set by the sensor supplier, they are often far too long. If you follows the rules given below, you should not have any problems,

**Note.** One of the most common reason for lousy process control is too heavy filtering. If your control starts swinging, check the filter settings first and reduce the filter times if they are too long.

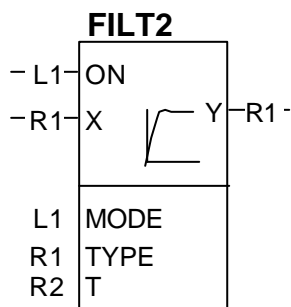
### Noise filter.

The *noise filter* is specified using the IOSET command. The purpose is to remove high-frequency noise. The filter is a simple first order filter, fixed for all modules using the same measurement. The noise filter time is normally set to about 50% of the basic cycling time in the application program .

The noise filter time can only be specified in fixed steps, see the IOSET command in chapter 7 for details.

### Regulator filtering.

The purpose of regulator filtering is too improve the signal used by the controller by removing disturbances which are not relevant or too fast for control. An efficient filter is the high-order FILT2 module in the function library:



This filter include 4<sup>th</sup> order Bessel, ITAE and Butterworth low-pass filters, which have very sharp frequency characteristics that will efficiently remove the higher frequencies with only minor distortion of the useful lower frequencies.

Select the filter time T as 50-80 % of the regulator sampling SAMP. The filter will then not affect the performance of the controller in a negative way. You should read the measurement signal about 10 times faster than the regulator sampling in your application program.

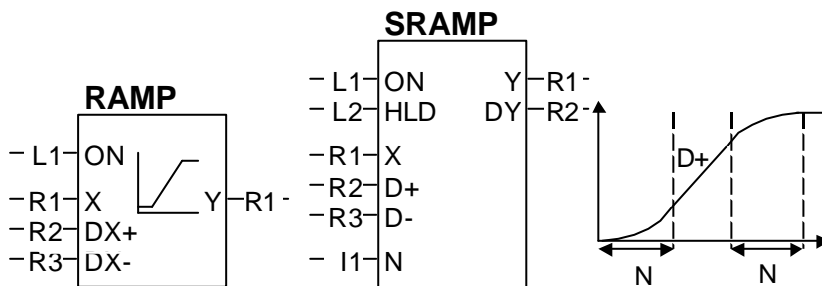
#### Rules of thumb

- ❑ Select the noise filter time about 50% of basic cycling time using the IOSET command.
- ❑ Use a high-order filter type FILT2 for the regulator filtering, e.g. a 4<sup>th</sup> order Bessel filter (TYPE=6).
- ❑ Select the filter time T in the filter as 50%-80% of the regulator sampling time determined by SAMP.
- ❑ If necessary, check the filter time in the sensor...

## Setpoint (SP)

In the STREGX2 controller, the regulator responses for disturbances and setpoint changes are separated, see section 9.2. This means that you can separately shape the setpoint response e.g. to avoid overshoots and violance of process limits without having to make the control slower.

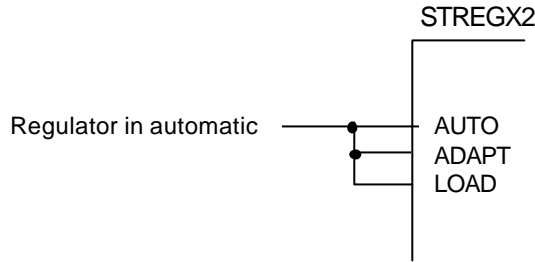
An efficient way to design the step response is often to use the ramp RAMP or the S-shaped ramp SRAMP modules to form the setpoint value entering the SP input:



Overshoots may be reduced by filtering the setpoint using a FILT module before entering the SP input.

## Man/auto switching (AUTO, ADAPT, LOAD)

It is common to connect a regulator AUTO signal to the ADAPT and LOAD inputs as well:



This means that the saved parameter set determined by the input MODEL will be loaded each time the regulator is switched to automatic. Moreover, the manual control input UE will be transferred to the regulator output U each block sample in manual mode since the adaption is switched off.

### Controller response (SAMP, POLE)

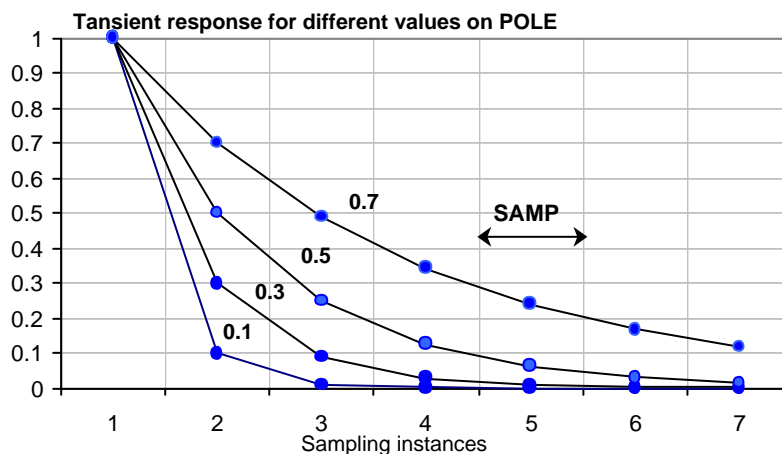
#### Default values

SAMP	Select 30-60% of desired response time
POLE	0.5

#### The feedback response.

The STREGX2 module contains an internal sampling SAMP which samples the regulator relative to the period of the function block. SAMP is the major parameter by which the user specifies the desired feedback response.

If the controller is specified to do feedback control (MODE=0,1,2), the closed loop transient is determined by the values of SAMP and POLE as



The *estimated* feedback response will be

POLE=0.5	The closed-loop settling time is about 4 x SAMP
----------	---

POLE=0.7	The closed-loop steeling time is about 7 x SAMP
----------	---

The *actual* feedback response is normally fairly close to the estimated value, but will depend on the dynamics of the specific process.

A good policy is to fix the POLE at 0.5-0.7 and then optimize the control response by changing the sampling time SAMP.

Start with a rather long sampling time so that the loop is sure to be stable. A normal choice could be the open-loop response time to about 50% of the steady-state value. The initial setting is not very crucial since the controller will converge for nearly any SAMP value which is not too small. The sampling-time is then decreased until the control performance is satisfactory or no longer improved. After each change, you should wait 30-40 regulator samples for the controller to be reasonably well tuned to the new setting.

In case the process contains a dominant *time delay*, choose the SAMP parameter so that sampling is made 1-3 times per delay. Faster sampling will normally result in only minor improvements. A first choice at start-up could be SAMP = time delay.

If the SAMP value is too small, the controller may try to exceed the physical limits of the process which leads to “nervous” control action and maybe occasional unstabilities. If this happens, the SAMP value should be increased.

#### Rules of thumb

- ❑ Select the POLE parameters as default, i.e. POLE =0.5.
- ❑ Select the initial value of SAMP equal to the open-loop response time to about 50% . The initial selection is not very crucial as long as it is not too small.
- ❑ In case there is a dominant time-delay in the process, select the initial value of SAMP = time delay.
- ❑ Decrease the value of SAMP to improve control performance.
- ❑ Increase the value of SAMP if the control action becomes too heavy or occasional unstabilities occur.

#### [The feedforward response.](#)

If the controller is a *pure feedforward* regulator (MODE =3,4), the SAMP value may be chosen more freely, since there is no risk for unstabilities as in the feedback case. To design the feedforward response, you must select the SAMP value so small that the disturbance you want to remove can be identified in the feedforward signal. You must also consider the feedforward *time window* (see above) since the controller must start control action *in advance* to make the control response effective relative to the process dynamics.

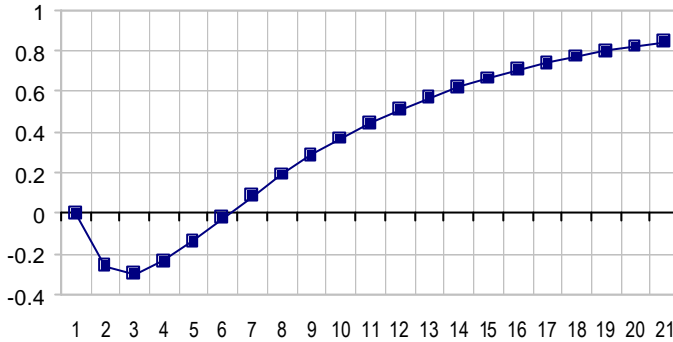
**Rules of thumb**

- ❑ Select SAMP small enough to incorporate the disturbance
- ❑ Design the feedforward so that the feedforward is within the time window..

See also "Model feedforward" above

### Non-minimum phase processes

Non-minimum phase behaviour is fairly common in industrial processes. A non-minimum phase process often responds in the wrong direction as is indicated in the figure below. More precisely, a non-minimum phase process is a process whose system inverse is unstable, i.e. unlimited control energy is needed to achieve ideal control.



For a non-minimum system, there are strict theoretical *limits in the control accuracy*. There exists no control that can improve the control accuracy beyond certain limits. For instance, in the example above, it is no meaning to try to compensate for the undershoot. This will only result in an unstable system or excess control amplitudes.

A non-minimum phase behaviour is often due to physical properties within the process. However, it may also be the result of a bad control structure where interacting control loops “counteract” each other. If this is the case, try to restructure your control loops.

#### **Rules of thumb**

- ❑ Select the SAMP parameter so large that the value exceeds the undershoot in the step response. In the example above, the SAMP parameter value should be larger than 6.
- ❑ The potential of improving the control accuracy is very limited in a non-minimum phase system using any type of control. If the process is non-minimum phase, you may have to accept slow closed-loop control.

### **Protective network (RESU, RESY, UMAX, UMIN)**

The STREGX2 parameters RESU, RESY, UMAX and UMIN are used by the protective network to set appropriate limits on the adapting part so that the controller is not degrading with time. The values are normally not crucial, but indicates for the controller the process limits to consider.

The first time you switch on the STREGX2 controller in a new process, you may deactivate the protective network so that the adapting process is active all the time until the controller is

reasonably well tuned to the process. You deactivate the protection by setting low values to the RESY and RESU parameters.

After the first start-up is made, one should activate the protection by setting appropriate value to RESU and RESY.

**Rules of thumb:**

- Set RESY = 0.025-0.1 % of total measurement range
- Set RESU = 0.025-0.1 % of total control range
- Use the lowest values at commissioning.

### Control limits (HI, LO,DUP,DUM)

Note the difference between the parameters UMAX, UMIN, defining the fixed control range and the inputs HI, LO defining limits of the control value that can be changed during running.

All limitations of the control value should be made directly in the STREGX2 module. No limitation should be made outside the module in the application program. If there are external limits in the process or in the actuator, the control limits of the STREGX2 module should be set so that the external limits are not violated.

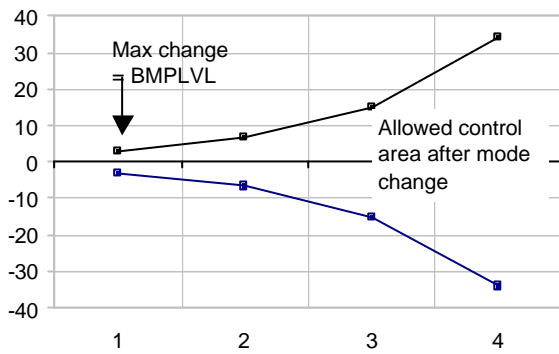
Any external limit in control action not handled by the STREGX2 module itself will easily result in oscillations or “wind-up” phenomena. This is true for most regulators except very simple regulator types which contain no internal memory such as P-regulators.

**Rules of thumb:**

- Use HI, LO, DUP, DUM for all limitation of the control value. No limitation should be done outside the module.
- If there are limits in the actuator or in the process itself, select HI,LO,DUP,DUM so that the control value does not violate those limits.
- Use DUP, DUM to prevent excessive control the first time the regulator is switched on.

### Mode switching (BMPLVL)

The BMPLVL parameter specifies the immediate maximal change in the control value after a mode switching AUTO: 0 to 1 or ON: 0 to 1. In the first step after switching, the control value is allowed to change at most BMPLV. In each step thereafter, the allowed change in the control value is multiplied by 1.25 until the value exceeds the values of the DUP and DUM inputs:

**Rules of thumb:**

- Set a very low value on BMPLVL during the first start-up of the regulator to protect the process.
- Then increase BMPLVL to the desired level.

**Adaption window (W)**

The adapting process must be performed in a slower time-scale than the regulation. The user can specify the time-scale by the adaption window ( $W$ ). The parameter  $W$  determines the speed by which the adaptive controller can change its control parameters. Normally, it takes about  $W$  regulator samples for the controller to retune itself after a change in the process dynamics such as e.g. a gain change.

The choice of  $W$  also depends on value of the regulator sampling SAMP. If you have a very small SAMP value, the  $W$  value should be increased to cover the same time range. Choose  $W$  in the area 50-2000 (corresponding to forgetting factors in the range 0.98-0.9995). The lower value is chosen if the sampling-time is long and/or you want to have fast adapting and the higher value is chosen if the sampling-time is short and/or you want the adapting to be slow.

**Rules of thumb:**

- Set  $W=100-200$
- Select always the lower value (100) at first start-up.



## Start-up procedure for STREGX2 module

### Preparations:

1. Choose parameter values according to the default or your own choice according to above.
2. Set parameters BMPLVL, DUP and DUM to small values in order to prevent excess control actions at start-up. If needed, you can also limit the control amplitudes with the HI and LO inputs.
3. Deactivate the protective network by setting low values to RESU and RESY so that the adapting process will run all the time.
4. Do not forget to specify a proper MODEL number for the model to be saved. Allowed numbers are 1,2...100.
5. Set the initial gain KINIT = 0.1.
6. Switch on the regulator. ON, AUTO and ADAPT must all be set to 1.

### Adjustments

1. If the control action is too large at start-up, decrease KINIT and/or decrease BMPLVL, DUP and DUM.
2. If possible, make a few small changes in the setpoint to make the adaption faster.
3. If the regulator has difficulties in getting into a stable behaviour increase SAMP (30-50% at a time). Wait at least 30-40 regulator samples before you make the next change
4. If the performance of the regulator is to be improved, decrease SAMP (30-50% at a time). Wait at least 30-40 regulator samples before you make the next change.
5. Be sure to save a model with the COMMAND DUMP command or by using the DUMP input fairly often in the beginning so that you don't have to start anew each time.

### Afterwards:

1. Set normal values on all parameters.
2. Be sure that you have saved the controller setting. This setting will be loaded at next cold start or by changing the value on the LOAD input.

