# arx

Estimate parameters of ARX or AR model using least squares

## Syntax

```
sys = arx(data,[na nb nk])
sys = arx(data,[na nb nk],Name,Value)
sys = arx(data,[na nb nk],__,opt)
```

## Description

> **Note** `arx` does not support continuous-time estimations. Use `tfest` instead.

`sys = arx(data,[na nb nk])` returns an ARX structure polynomial model, `sys`, with estimated parameters and covariances (parameter uncertainties) using the least-squares method and specified `orders`.

`sys = arx(data,[na nb nk],Name,Value)` estimates a polynomial model with additional options specified by one or more `Name,Value` pair arguments.

`sys = arx(data,[na nb nk],__,opt)` specifies estimation options that configure the estimation objective, initial conditions and handle input/output data offsets.

## Input Arguments

`data`

Estimation data.

Specify `data` as an `iddata` object, an `frd` object, or an `idfrd` frequency-response-data object.

`[na nb nk]`

Polynomial orders.

`[na nb nk]` define the polynomial orders of an ARX model.

- `na` — Order of the polynomial $A(q)$.

  Specify `na` as an $Ny$-by-$Ny$ matrix of nonnegative integers. $Ny$ is the number of outputs.
- `nb` — Order of the polynomial $B(q) + 1$.

  `nb` is an $Ny$-by-$Nu$ matrix of nonnegative integers. $Ny$ is the number of outputs and $Nu$ is the number of inputs.
- `nk` — Input-output delay expressed as fixed leading zeros of the $B$ polynomial.

  Specify `nk` as an $Ny$-by-$Nu$ matrix of nonnegative integers. $Ny$ is the number of outputs and $Nu$ is the number of inputs.

`opt`

Estimation options.

`opt` is an options set that specifies estimation options, including:

- input/output data offsets
- output weight

Use `arxOptions` to create the options set.

### Name-Value Pair Arguments

Specify optional comma-separated pairs of `Name,Value` arguments, where `Name` is the argument name and `Value` is the corresponding value. `Name` must appear inside single quotes (`' '`). You can specify several name and value pair arguments in any order as `Name1,Value1,...,NameN,ValueN`.

| | |
|---|---|
| `'InputDelay'` | Input delays. `InputDelay` is a numeric vector specifying a time delay for each input channel. Specify input delays in integer multiples of the sampling period `Ts`. For example, `InputDelay = 3` means a delay of three sampling periods.<br><br>For a system with `Nu` inputs, set `InputDelay` to an `Nu`-by-1 vector, where each entry is a numerical value representing the input delay for the corresponding input channel. You can also set `InputDelay` to a scalar value to apply the same delay to all channels.<br><br>**Default:** 0 for all input channels |
| `'ioDelay'` | Transport delays. `ioDelay` is a numeric array specifying a separate transport delay for each input/output pair.<br><br>Specify transport delays as integers denoting delay of a multiple of the sampling period `Ts`.<br><br>For a MIMO system with `Ny` outputs and `Nu` inputs, set `ioDelay` to a `Ny`-by-`Nu` array, where each entry is a numerical value representing the transport delay for the corresponding input/output pair. You can also set `ioDelay` to a scalar value to apply the same delay to all input/output pairs. Useful as a replacement for the `nk` order, you can factor out `max(nk-1,0)` lags as the `ioDelay` value.<br><br>**Default:** 0 for all input/output pairs |
| `'IntegrateNoise'` | Logical vector specifying integrators in the noise channel.<br><br>`IntegrateNoise` is a logical vector of length *Ny*, where *Ny* is the number of outputs.<br><br>Setting `IntegrateNoise` to `true` for a particular output results in the ARIX model:<br><br>$$A(q)y(t) = B(q)u(t-nk) + \frac{1}{1-q^{-1}}e(t)$$<br><br>Where, $\frac{1}{1-q^{-1}}$ is the integrator in the noise channel, *e*(*t*).<br><br>**Default:** `false(Ny,1)` (*Ny* is the number of outputs.) |

## Output Arguments

| | |
|---|---|
| `sys` | Identified ARX structure polynomial model.<br><br>`sys` is a discrete-time `idpoly` model, which encapsulates the estimated *A* and *B* polynomials and the parameter covariance information. |

## Definitions

### ARX structure

`arx` estimates the parameters of the ARX model structure:

$$y(t) + a_1 y(t-1) + \ldots + a_{na} y(t-na) =$$
$$b_1 u(t-nk) + \ldots + b_{nb} u(t-nb-nk+1) + e(t)$$

The parameters `na` and `nb` are the orders of the ARX model, and `nk` is the delay.

- $y(t)$ — Output at time $t$.
- $n_a$ — Number of poles.
- $n_b$ — Number of zeroes plus 1.
- $n_k$ — Number of input samples that occur before the input affects the output, also called the *dead time* in the system.
- $y(t-1)\ldots y(t-n_a)$ — Previous outputs on which the current output depends.
- $u(t-n_k)\ldots u(t-n_k-n_b+1)$ — Previous and delayed inputs on which the current output depends.
- $e(t-1)\ldots e(t-n_c)$ — White-noise disturbance value.

A more compact way to write the difference equation is

$$A(q)y(t) = B(q)u(t-n_k) + e(t)$$

*q* is the delay operator. Specifically,

$$A(q) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$$

$$B(q) = b_1 + b_2 q^{-1} + \ldots + b_{n_b} q^{-n_b+1}$$

### Time Series Models

For time-series data that contains no inputs, one output and `orders = na`, the model has AR structure of order `na`.

The AR model structure is

$$A(q)y(t) = e(t)$$

### Multiple Inputs and Single-Output Models

For multiple-input systems, `nb` and `nk` are row vectors where the `i`th element corresponds to the order and delay associated with the `i`th input.

$$y(t) + A_1 y(t-1) + A_2 y(t-2) + \ldots + A_{na} y(t-na) =$$
$$B_0 u(t) + B_1 u(t-1) + \ldots + B_{nb} u(t-nb) + e(t)$$

### Multi-Output Models

For models with multiple inputs and multiple outputs, `na`, `nb`, and `nk` contain one row for each output signal.

In the multiple-output case, `arx` minimizes the trace of the prediction error covariance matrix, or the norm

$$\sum_{t=1}^{N} e^T(t)e(t)$$

To transform this to an arbitrary quadratic norm using a weighting matrix `Lambda`

$$\sum_{t=1}^{N} e^T(t)\Lambda^{-1}e(t)$$

use the syntax

```
opt = arxOptions('OutputWeight', inv(lambda))
m = arx(data, orders, opt)
```

### Estimating Initial Conditions

For time-domain data, the signals are shifted such that unmeasured signals are never required in the predictors. Therefore, there is no need to estimate initial conditions.

For frequency-domain data, it might be necessary to adjust the data by initial conditions that support circular convolution.

Set the `InitialCondition` estimation option (see `arxOptions`) to one the following values:

- `'zero'` — No adjustment.
- `'estimate'` — Perform adjustment to the data by initial conditions that support circular convolution.
- `'auto'` — Automatically choose between `'zero'` and `'estimate'` based on the data.

## Examples

This example generates input data based on a specified ARX model, and then uses this data to estimate an ARX model.

```
A = [1  -1.5  0.7]; B = [0 1 0.5];
m0 = idpoly(A,B);
u = iddata([],idinput(300,'rbs'));
e = iddata([],randn(300,1));
y = sim(m0, [u e]);
z = [y,u];
m = arx(z,[2 2 1]);
```

## Algorithms

QR factorization solves the overdetermined set of linear equations that constitutes the least-squares estimation problem.

The regression matrix is formed so that only measured quantities are used (no fill-out with zeros). When the regression matrix is larger than `MaxSize`, data is segmented and QR factorization is performed iteratively on these data segments.

## See Also

`ar` | `armax` | `arxOptions` | `arxstruc` | `bj` | `iv4` | `n4sid` | `nlarx` | `oe`

## How To

- Using Linear Model for Nonlinear ARX Estimation

Was this topic helpful? Yes No