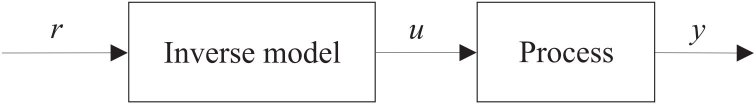
Considered Settings

- Fuzzy or neural model of the process available
(many of the presented techniques apply to other types of models as well)
- Based on the model, design a controller (off line)
- Use the model explicitly within a controller
- Model fixed or adaptive

Inverse Control (Feedforward)



Process model: $y(k+1) = f\big(\boldsymbol{x}(k), u(k)\big)$, where

$$\boldsymbol{x}(k) = \big[y(k), \ldots, y(k-n_y+1), u(k-1), \ldots, u(k-n_u+1)\big]^T$$

Controller: $u(k) = f^{-1}\big(\boldsymbol{x}(k), r(k+1)\big)$

When is Inverse-Model Control Applicable?

1. Process (model) is stable and invertible
2. The inverse model is stable
3. Process model is accurate (enough)
4. Little influence of disturbances
5. In combination with feedback techniques

How to invert $f(\cdot)$?

1. Numerically (general solution, but slow):

$$J(u(k)) = \left[r(k+1) - f\left(\boldsymbol{x}(k), u(k)\right) \right]^2$$

minimize w.r.t. $u(k)$

How to invert $f(\cdot)$?

1. Numerically (general solution, but slow):

$$J(u(k)) = \Big[r(k+1) - f\big(\boldsymbol{x}(k), u(k)\big)\Big]^2$$

   minimize w.r.t. $u(k)$

2. Analytically (for some special forms of $f(\cdot)$ only):
   - affine in $u(k)$
   - singleton fuzzy model

How to invert $f(\cdot)$?

1. Numerically (general solution, but slow):

$$J(u(k)) = \Big[r(k+1) - f\big(\boldsymbol{x}(k), u(k)\big)\Big]^2$$

   minimize w.r.t. $u(k)$

2. Analytically (for some special forms of $f(\cdot)$ only):
   - affine in $u(k)$
   - singleton fuzzy model
3. Construct inverse model directly from data

Inverse of an Affine Model

affine model:

$$y(k+1) = g\big(\boldsymbol{x}(k)\big) + h\big(\boldsymbol{x}(k)\big) \cdot u(k)$$

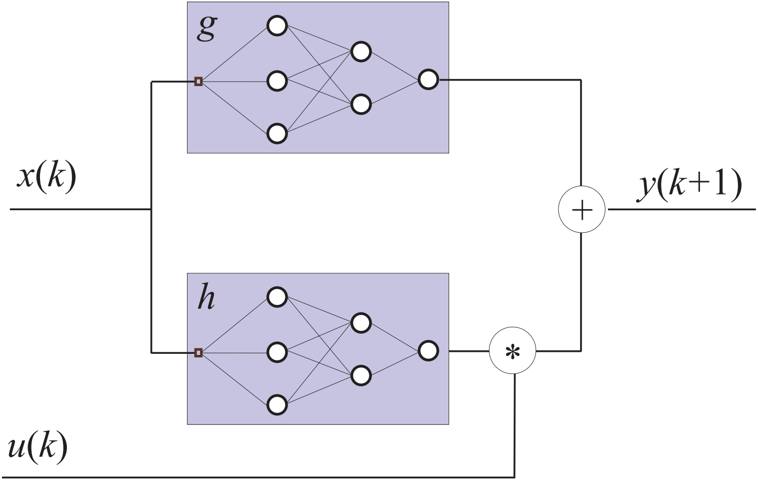
express $u(k)$:

$$u(k) = \frac{y(k+1) - g\big(\boldsymbol{x}(k)\big)}{h\big(\boldsymbol{x}(k)\big)}$$

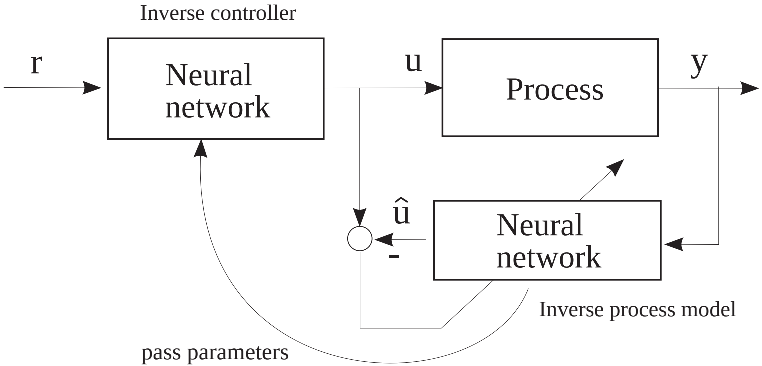
substitute $r(k+1)$ for $y(k+1)$

necessary condition $h(\boldsymbol{x}) \neq 0$ for all $\boldsymbol{x}$ of interest

Example: Affine Neural Network



Learning Inverse (Neural) Model



How to obtain $\boldsymbol{x}$?

inverse model: $u(k) = f^{-1}\big(\boldsymbol{x}(k), r(k+1)\big)$

1. Use the prediction model: $\hat{y}(k+1) = f\left(\hat{\boldsymbol{x}}(k), u(k)\right)$

   $$\hat{\boldsymbol{x}}(k) = \left[\hat{y}(k), \ldots, \hat{y}(k-n_y+1), u(k-1), \ldots, u(k-n_u+1)\right]^T$$

   Open-loop feedforward control

How to obtain $\boldsymbol{x}$?

inverse model: $u(k) = f^{-1}(\boldsymbol{x}(k), r(k+1))$

1. Use the prediction model: $\hat{y}(k+1) = f(\hat{\boldsymbol{x}}(k), u(k))$

   $$\hat{\boldsymbol{x}}(k) = [\hat{y}(k), \ldots, \hat{y}(k-n_y+1), u(k-1), \ldots, u(k-n_u+1)]^T$$

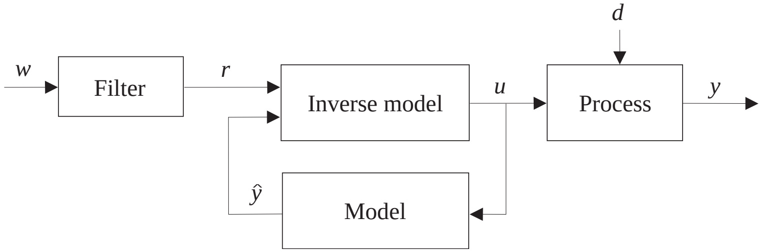
   Open-loop feedforward control

2. Use measured process output

   $$\boldsymbol{x}(k) = [y(k), \ldots, y(k-n_y+1), u(k-1), \ldots, u(k-n_u+1)]^T$$

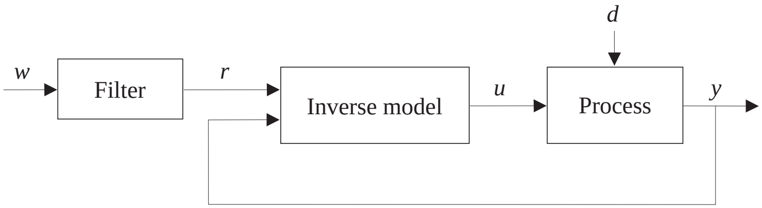
   Open-loop feedback control

Open-Loop Feedforward Control



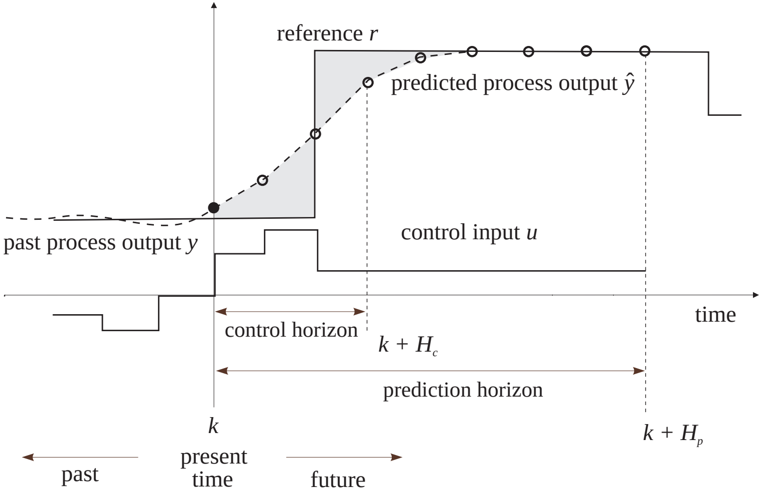
- Always stable (for stable processes)
- No way to compensate for disturbances

Open-Loop Feedback Control



- Can to some degree compensate disturbances
- Can become unstable

Model-Based Predictive Control



Objective Function and Constraints

$$J = \sum_{i=1}^{H_p} \left\| \boldsymbol{r}(k+i) - \hat{\boldsymbol{y}}(k+i) \right\|_{P_i}^2 + \sum_{i=1}^{Hc} \left\| \boldsymbol{u}(k+i-1) \right\|_{Q_i}^2$$

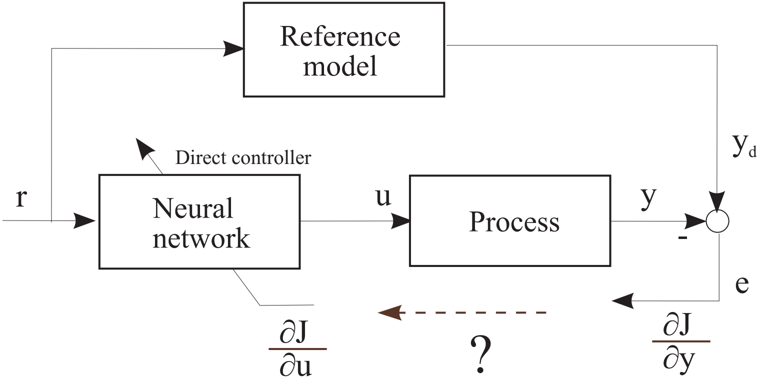
$$\hat{y}(k+1) = f\left(\hat{\boldsymbol{x}}(k), u(k)\right)$$

$$\begin{array}{ccccc}
\boldsymbol{u}^{\min} & \leq & \boldsymbol{u} & \leq & \boldsymbol{u}^{\max} \\
\Delta\boldsymbol{u}^{\min} & \leq & \Delta\boldsymbol{u} & \leq & \Delta\boldsymbol{u}^{\max} \\
\boldsymbol{y}^{\min} & \leq & \boldsymbol{y} & \leq & \boldsymbol{y}^{\max} \\
\Delta\boldsymbol{y}^{\min} & \leq & \Delta\boldsymbol{y} & \leq & \Delta\boldsymbol{y}^{\max}
\end{array}$$

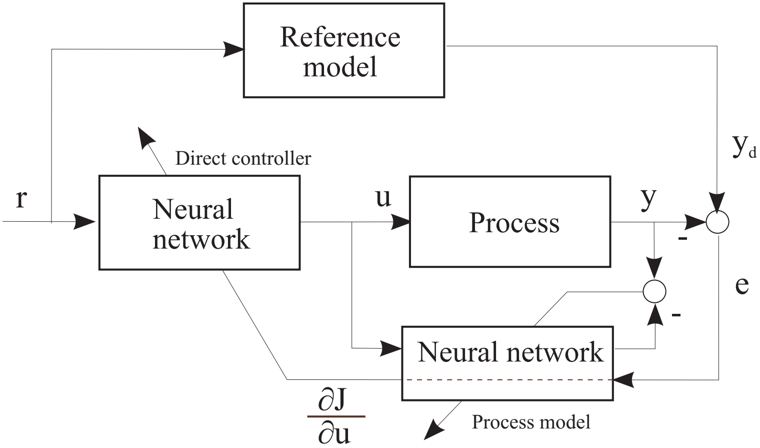
Adaptive Control

- Model-based techniques (use explicit process model):
    - model reference control through backpropagation
    - indirect adaptive control
- Model-free techniques (no explicit model used)
    - reinforcement learning

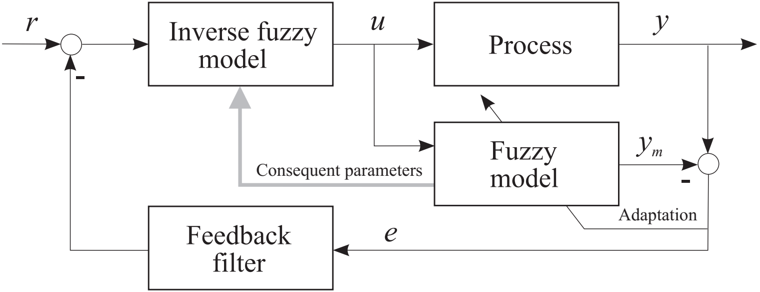
Model Reference Adaptive Neurocontrol



Model Reference Adaptive Neurocontrol



Indirect Adaptive Control



not only for fuzzy models, but also for affine NNs, etc.