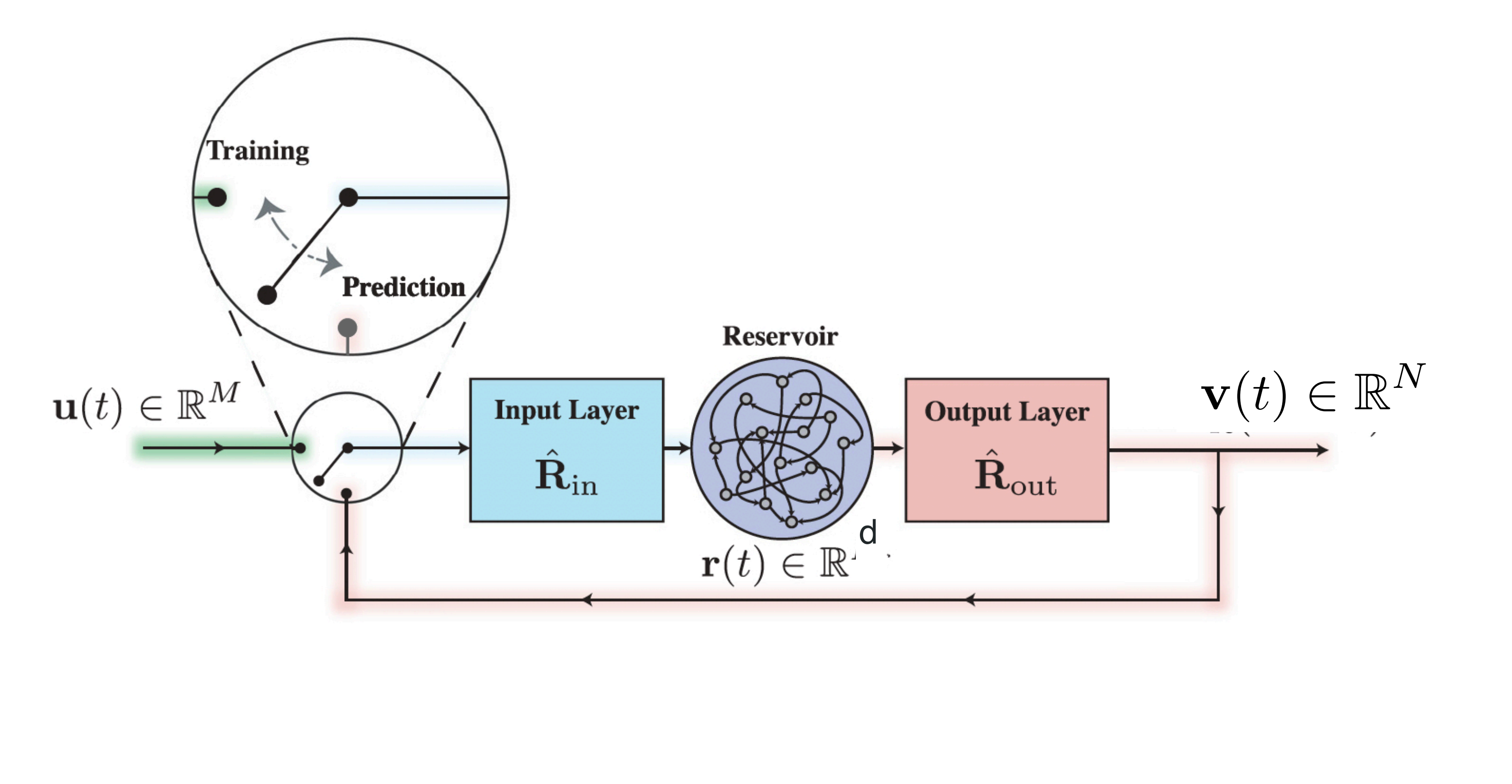
Reservoir Computing



Reservoir computing

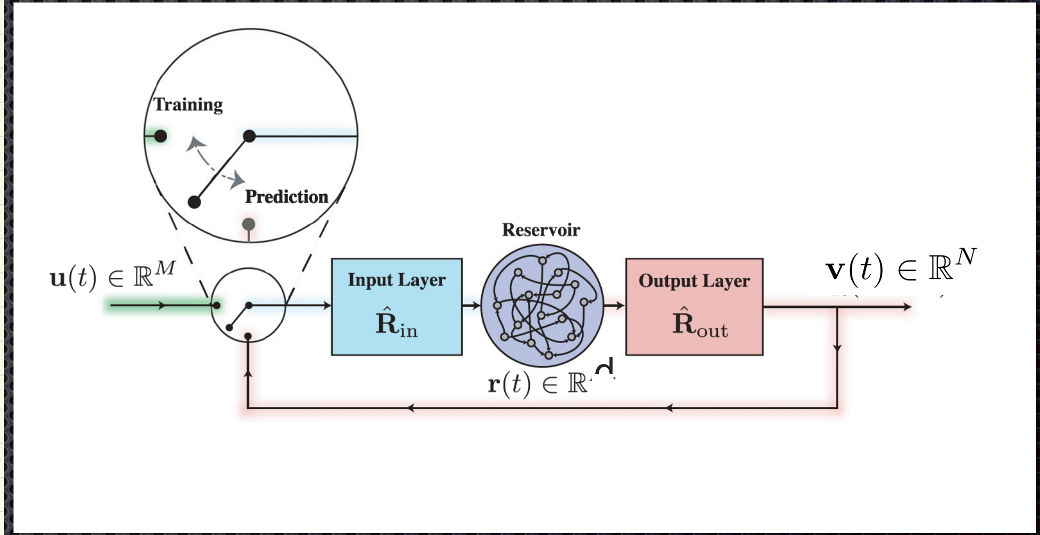
1.a) Input data $\underline{u}(t) \in \mathbb{R}^M$

$t \in [-T, 0]$

These data will be used to train the 'machine' for the purpose of making predictions for $t > 0$.

b) Reservoir computer

State vector $\underline{r}(t) \in \mathbb{R}^d$, with $d \gg M$. As shown below, the reservoir will be constructed as a dynamical system on a weighted random network.



(2)

c) Output data $\underline{v}(t) \in \mathbb{R}^N$

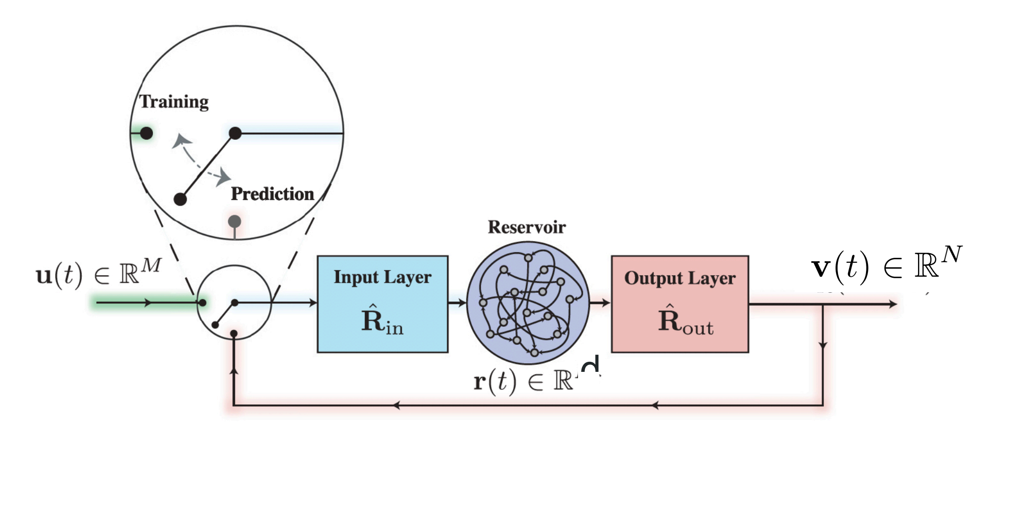
The desired output data

d) $\hat{R}_{in} : \mathbb{R}^M \to \mathbb{R}^d$ maps the input data to the reservoir state, hence

$$\underline{r}(t) = \hat{R}_{in}\left[\underline{u}(t)\right]$$

$\hat{R}_{out} : \mathbb{R}^d \to \mathbb{R}^N$ maps the reservoir state to the output, i.e.,

$$\underline{v}(t) = \hat{R}_{out}\left[\underline{r}(t)\right]$$



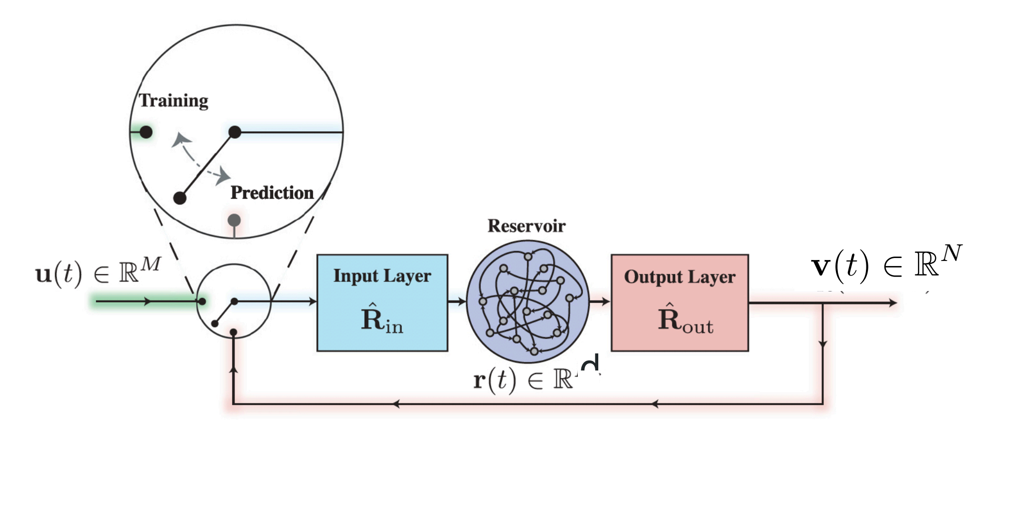
2. Reservoir dynamics.

The reservoir is often chosen as a random weigthed & directed network which is constructed in the following way (# nodes = d)

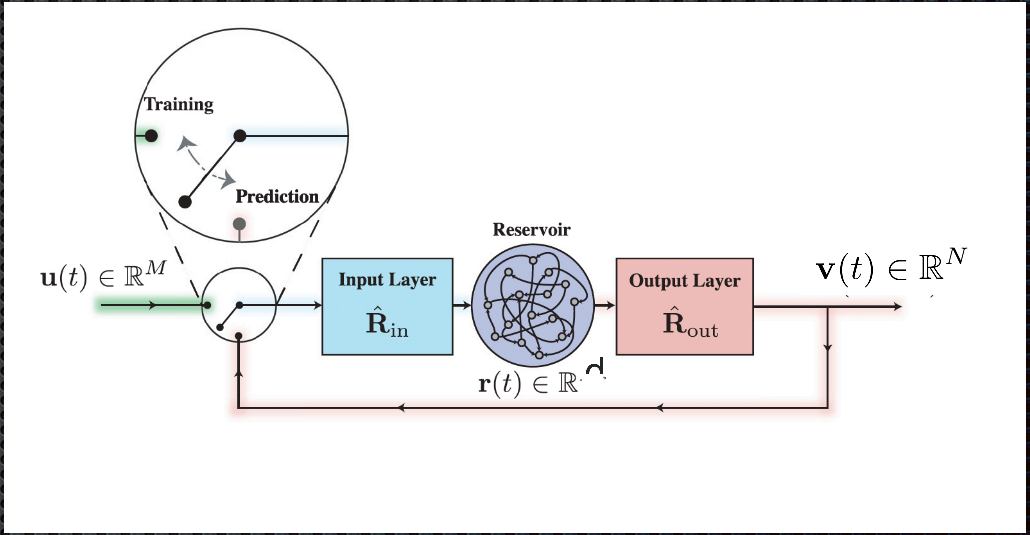
(i) Generate a directed ER (random) network with mean $\langle k_{in} \rangle = \langle k_{out} \rangle = \langle k \rangle$

(ii) Choose edge weigths from a $U[-1,1]$ (uniform) distribution.

(iii) Determine adjacency matrix $\tilde{A}$: $d \times d$



(iv) Rescale A with a constant c
($A = cA$) so that the norm of
the largest eigenvalue is equal
to ρ (the spectral radius).



Let $\hat{R}_{in}[\underline{u}(t)] = W_{in}\,\underline{u}(t)$

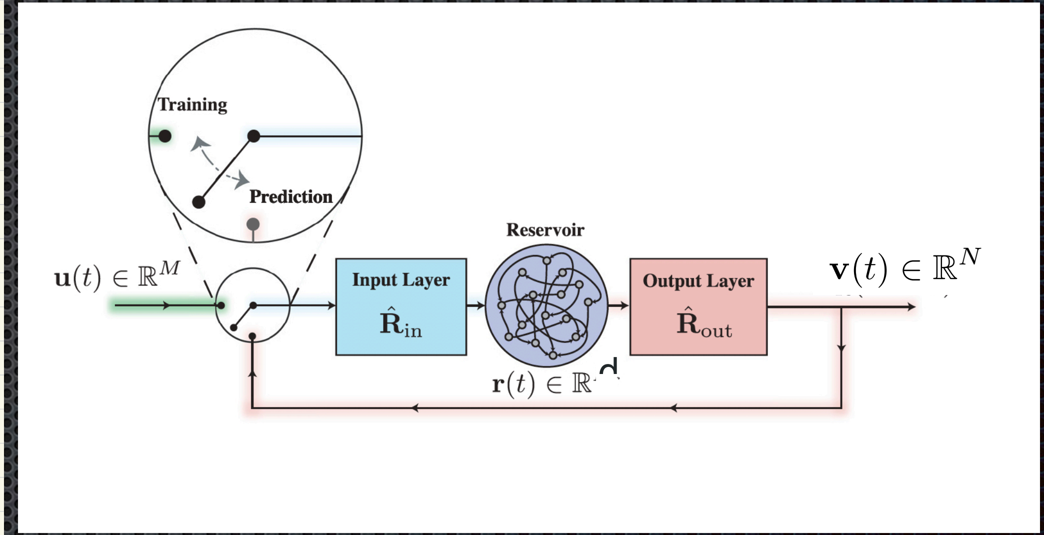
where $W_{in} : d \times M$, where each row has exactly one element $\in U[-\sigma, \sigma]$. The discrete dynamics on the network is then given by

component wise

$$\underline{r}(t+\Delta t) = \tanh\left(A\,\underline{r}(t) + W_{in}\,\underline{u}(t)\right)$$

Note that we introduced already several hyperparameters:

- d : # nodes
- σ : coupling constant in W_{in}
- $\langle k \rangle$: mean degree
- c or ρ : spectral radius of A



3. Training of the network

- Over the period $[-T, 0]$, the reservoir evolves according to

$$\underline{r}(t) = \tanh\left(A\,\underline{r}(t) + W_{in}\,\underline{u}(t) \right)$$

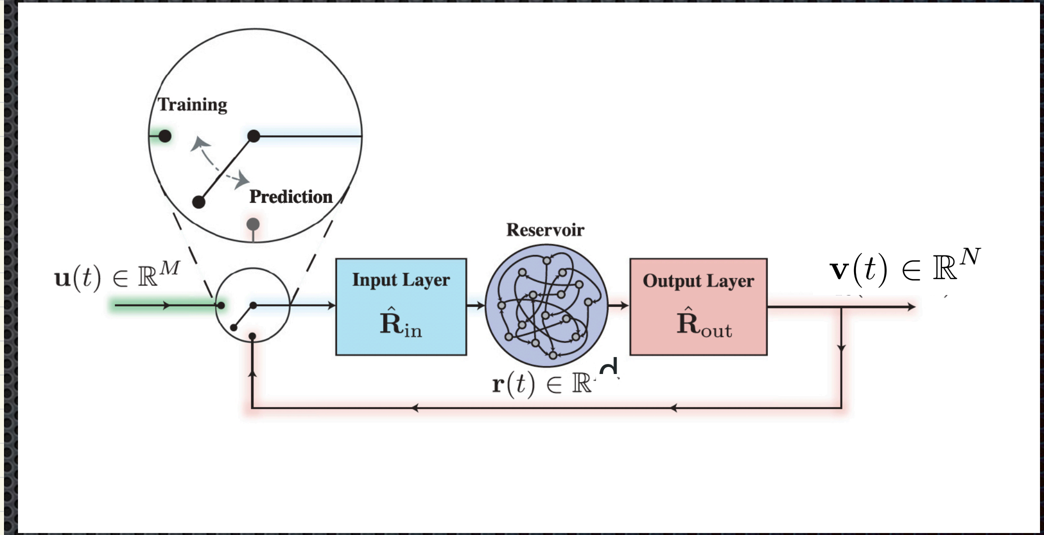
so we have $\frac{T}{\Delta t} = n$ vectors

$$(\underline{r}_1, \ldots, \underline{r}_n).$$

- Sometimes the following is used

$$r_j^* = \begin{cases} r_j, & j \text{ odd} \\ r_j^2, & j \text{ even} \end{cases}$$

but we will omit this here



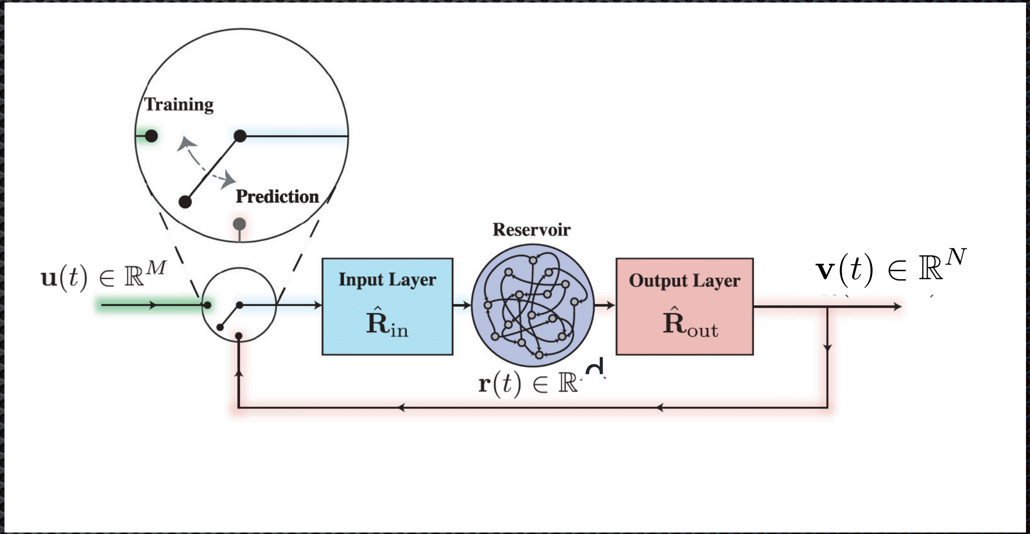
- Define W_{out} as an $M \times d$ matrix of weights, and define ($j = 1, \dots, n$):

$$\underline{v}_j = \hat{R}_{out}(\underline{r}_j) = W_{out}\,\underline{r}_j$$

- Let the input vectors $\underline{u}(t)$ over the discrete times be indicated by $(\underline{u}_1, \dots, \underline{u}_n)$, then the W^*_{out} is determined from

$$W^*_{out} = \min_{W} \left\{ \frac{1}{2} \sum_{j=1}^{n} \| \underline{u}_j - \underline{v}_j \|^2 + \frac{\beta}{2} \| W_{out} \|^2 \right\}$$

where $\|W_{out}\|^2$ is the matrix norm (sum over all squares).



4. Solution of the optimization problem.

<u>Example</u>: $M = 2$, $d = 3$, $n = 1$

$$W_{out} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{pmatrix}$$

$$\underline{r}(t) = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}(t) \; ; \; \begin{cases} \underline{u}(t) = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}(t) \\ \underline{v}(t) = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}(t) \end{cases}$$

Hence:

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} w_{11} r_1 + w_{12} r_2 + w_{13} r_3 \\ w_{21} r_1 + w_{22} r_2 + w_{23} r_3 \end{pmatrix}$$

which can be written as

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \underbrace{\begin{pmatrix} r_1 & r_2 & r_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_1 & r_2 & r_3 \end{pmatrix}}_{X} \overbrace{\begin{pmatrix} w_{11} \\ w_{12} \\ w_{13} \\ w_{21} \\ w_{22} \\ w_{23} \end{pmatrix}}^{\underline{w}}$$

Cost function is in this case :

$$J(\underline{w}) = \frac{1}{2}\left(\underline{u} - \underline{v}\right)^2 + \beta/_2 \|w\|^2$$

$$= \frac{1}{2}\left(\underline{u} - X\,\underline{w}\right)^2 + \beta/_2 \|w\|^2$$

Minimum , need again ∇J

$$\underset{6\times 1}{\nabla J} = -\underset{6\times 2}{X^T}\left(\underset{2\times 1}{\underline{u}} - \underset{2\times 6}{X}\,\underset{6\times 1}{\underline{w}}\right) + \beta I \underline{w}$$

$\underline{w}_* = \min_{w} J(\underline{w})$ determined from $\nabla J = 0$ $\longrightarrow$

$$-X^T(\underline{u} - X\underline{w}) + \beta I \underline{w} = 0$$

$$\longrightarrow \quad \underline{w} = (\beta \underline{I} + X^T X)^{-1} X^T \underline{u}$$

This is just a linear regression problem. If $n > 1$, then

$$X = \begin{array}{cc} \overset{d}{\longleftrightarrow} & \\ \begin{pmatrix} T_1^T & 0 & 0 \\ 0 & T_1^T & 0 \\ 0 & 0 & T_1^T \\ & \vdots & \\ T_n^T & 0 & 0 \\ 0 & T_n^T & 0 \\ 0 & 0 & T_n^T \end{pmatrix} & \updownarrow M \end{array} \qquad \underline{T}_j : d \times 1$$

<u>When</u> $\beta = 0$:

$$\underline{w} = \underbrace{\left(X^T X\right)^{-1} X^T}_{\text{Moore - Penrose inverse}} \underline{y}$$

(12)

5. Prediction

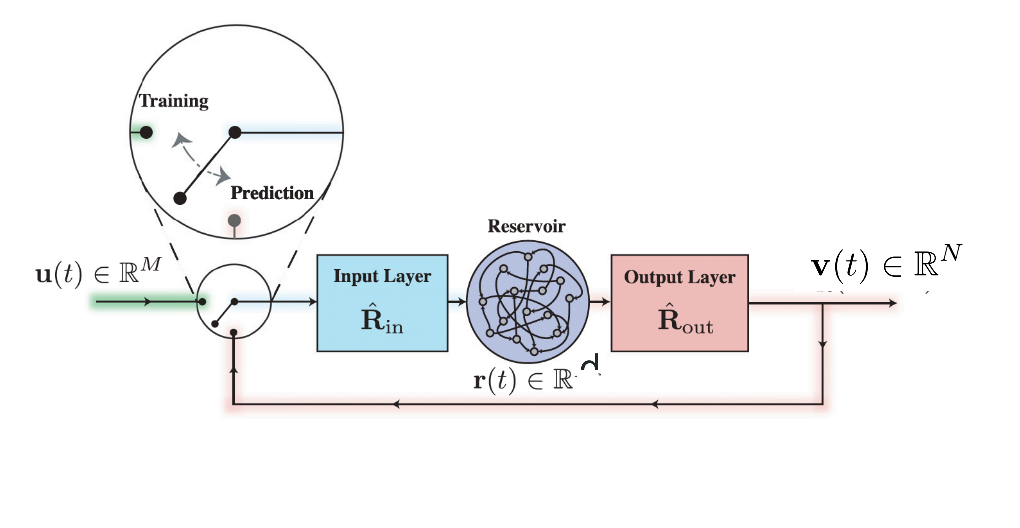
Once the weights are determined, forecasts can be made for $t > 0$ using :

$$\begin{cases} \underline{r}(t+\Delta t) = \tanh\left(A\,\underline{r}(t) + W_{in}\,\underline{v}(t)\right) \\ \underline{v}(t) = W_{out}\,\underline{r}(t) \end{cases}$$

- Initial conditions :

  - $\underline{u}(0)$ is the last input of the training set.
  - we want to choose $\underline{r}(0)$ such that $\underline{v}(0) = \underline{r}(0)$

    hence $\underline{u}(0) = W_{out}\,\underline{r}(0)$

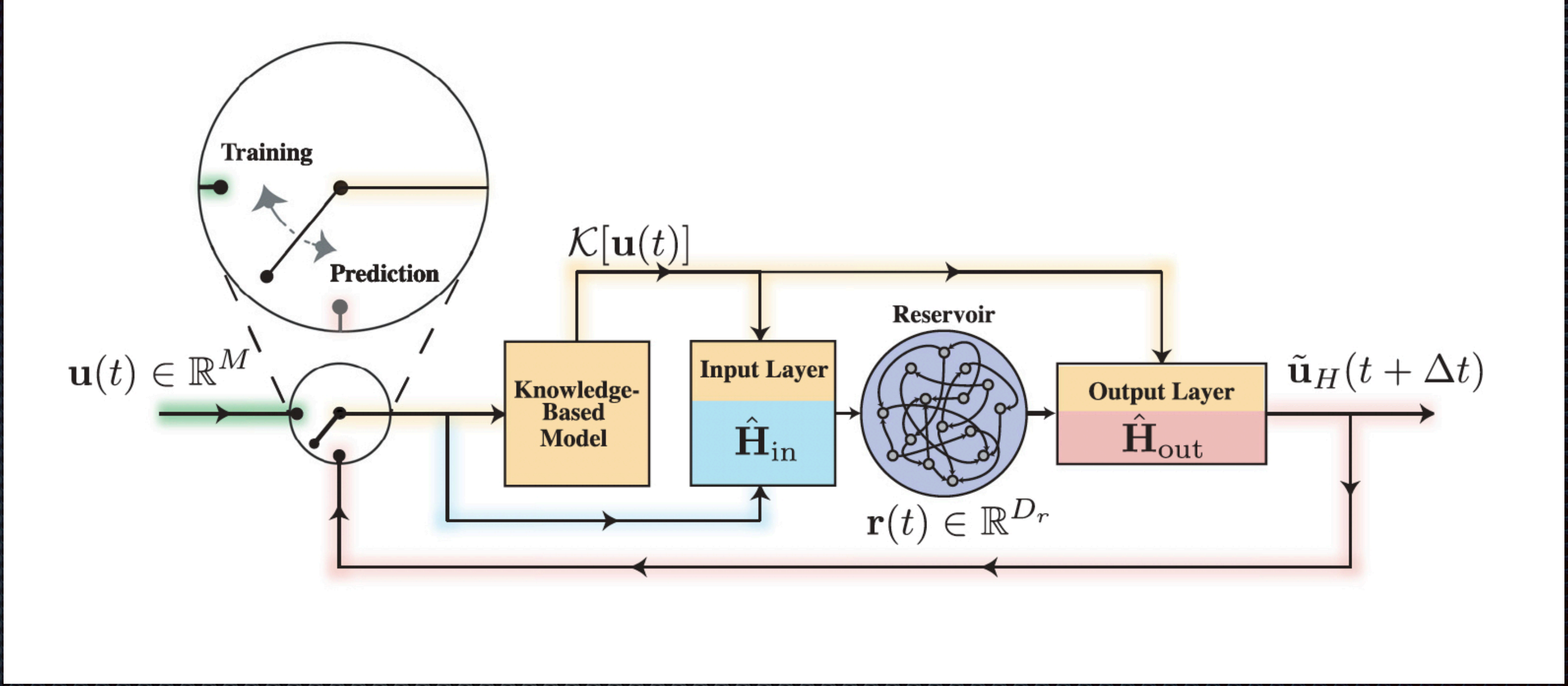


Multiplying by W_{in}, we find

$$W_{in}\, \underline{u}(0) = \underbrace{W_{in} W_{out}}_{\psi}\, \underline{r}(0)$$

$$\longrightarrow \underbrace{\underline{r}(0)}_{d\times 1} = (\underbrace{W_{in}}_{d\times M} \underbrace{W_{out}}_{M\times d})^{-1} \underbrace{W_{in}}_{d\times M} \underbrace{\underline{u}(0)}_{M\times 1}$$

Hybrid Reservoir Computing



Exercise B5