3.1 Architectures for Approximation.

"Approximate cost-to-go function using best-fit parameters".

3.1.1 Overview of Approximating Architectures.

$$J(i) \approx \tilde{J}(i,r)$$

Linear -

$$\tilde{J}(i,r) = \sum_{k=0}^{K} r(k) \underbrace{\phi_k(i)}_{\text{basis functions}}.$$

$$\tilde{J}(i,r) = r(0) + \sum_{k=1}^{K} r(k)\,\phi_k(i).$$

Pose a least squares problem,

$$\sum_i \Big(J(i) - \sum_k r(k)\phi_k(i) \Big)^2.$$

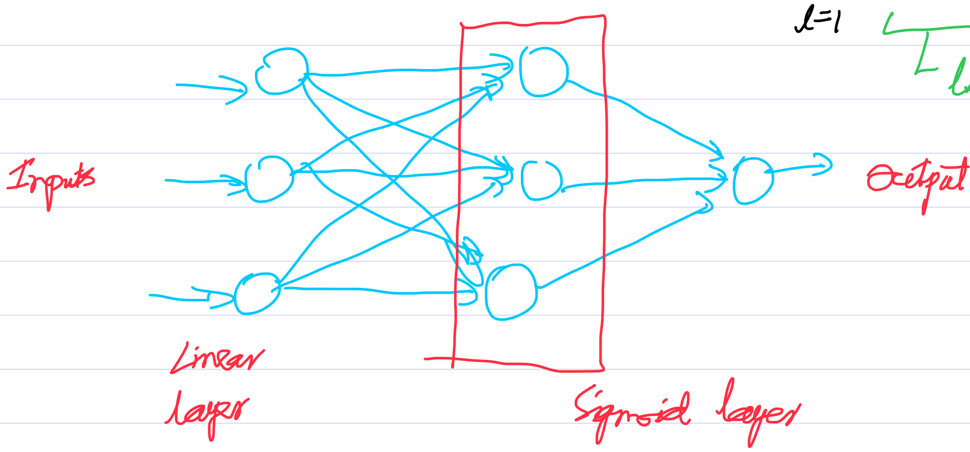
Nonlinear -

$$\sum_i \big(J(i) - \tilde{J}(i,r) \big)^2$$

- Not reduced to a linear algebraic problem.

Multilayer Perceptrons -

$$\underbrace{\sum_{l=1}^{L} r(k,l) \overbrace{x_l(i)}^{\text{inputs}}}_{\text{linear layer}}, \; k=1, \ldots K$$



$$\sigma\left(\sum_{\ell} r(k,\ell)\, x_\ell(i)\right),$$

The final output is $\tilde{J}(i,r) = \sum_{k} r(k)\, \sigma\left(\sum_{\ell} r(k,\ell)\, x_\ell(i)\right).$

"Train MLPs using backpropagation".

Two step process

(a) Use a forward pass to calculate sequentially the outputs of the linear layers.

(b) Use a backward pass to calculate sequentially the derivatives.

3.1.2 Features.

Break the complexity of network into smaller modules.

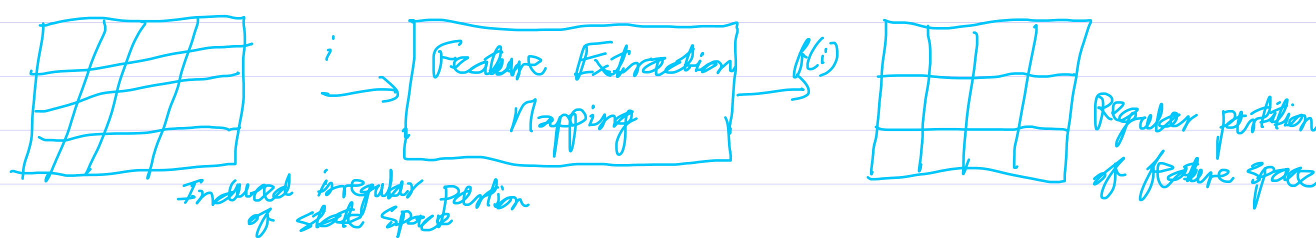
feature $f_k : S \to R$, $f(i) = (f_1(i), f_2(i), \ldots f_k(i)).$

Approximate $\tilde{J}(f(i), r)$.

$$\sum_i \left(J(i) - \sum_k r(k)\, \phi_k(f(i))\right)^2.$$

3.1.3 Partitioning.

Construct sophisticated architectures by partitioning the state-space and construct an approximation for each partition.



$$\tilde{J}(i;r) = \hat{J}(i;\hat{r}) + \sum_{m=1}^{M} \sum_{k=1}^{K_m} r_m(k)\, \phi_{k,m}(i).$$

One can minimize distance using,

$$\sum_i \left(J(i) - \hat{J}(i;r) - \sum_{m=1}^{M} \sum_{k=1}^{K_m} r_m(k)\, \phi_{k,m}(i) \right)^2.$$

3.1.4 Using Heuristic Policies to Construct Features

Consider a set of heuristic policies μ_k so that we can fairly approximate $\tilde{J}^{\mu_k}(i)$,

$$\tilde{J}(i;r) = w_0(i;r_0) + \sum_{k=1}^{K} w_k(i;r_k)\, \tilde{J}^{\mu_k}(i).$$

The key idea is that by using different weights $w_k(i;r_k)$ at different states i, we may be able to learn how to identify the most promising heuristic policy.

Suppose we have an M-dimensional feature vector,

$$f(i) = (f_1(i), \dots f_M(i)).$$

Then we may consider a linear parameterization $w_k(i;r_k)$,

$$w_k(i;r_k) = r_k(0) + \sum_{m=1}^{M} r_k(m)\, f_m(i).$$

The architecture can be written as,

$$\tilde{J}(i;r) = r_0(0) + \sum_{m=1}^{M} r_0(m)\, f_m(i) + \sum_{k=1}^{K} r_k(0)\, \tilde{J}^{\mu_k}(i)$$

$$+ \sum_{m=1}^{M} \sum_{k=1}^{K} r_k(m)\, f_m(i)\, \tilde{J}^{\mu_k}(i)$$