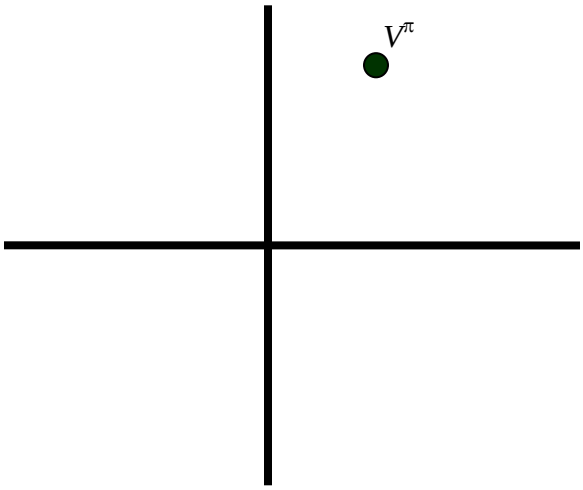
# Example for Function Approximation

Consider an MDP with 2 states: *A* and *B*. We want to build a linear function approximator that can represent a value function over these two states. Assume that there exists some policy $\pi$ under which $V^{\pi}(A) = 2$ and $V^{\pi}(B)=4$. This particular "target" value function is a point in the space $\mathbb{R}^2$ namely (2,4) as shown below:

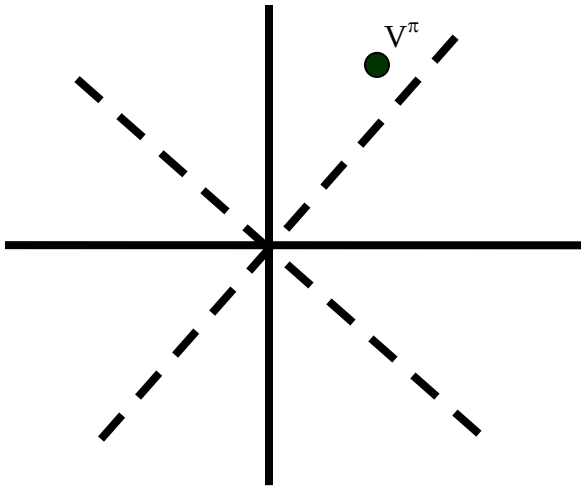



If we use the standard basis functions as the features for our linear function approximator, we end up with the following representation for our states: $\Phi(A)=(1,0)$, and $\Phi(B) = (0,1)$. To use the notation from class, we should say: $\phi_1(A) =1$, $\phi_2(A)=0$; $\phi_1(B)=0$, $\phi_2(B)=1$. The output of the function approximator is then $W^T\Phi(.)$ $[=w_1\phi_1(.)+w_2\phi_2(.)]$, for some weight vector $W = (w_1,w_2)$. To represent the above value function then, $W = (2,4)$. As can be seen, the standard basis can be used to represent any point in the space $\mathbb{R}^2$, and hence any value function over two states.

Suppose we use the following features to represent the states: $\Phi(A)=(1,-1)$, and $\Phi(B) = (1,1)$. This representation also constitutes a basis and hence should be able to represent the given value function. What would the values of the weights be? We need to solve the following set of linear equations:

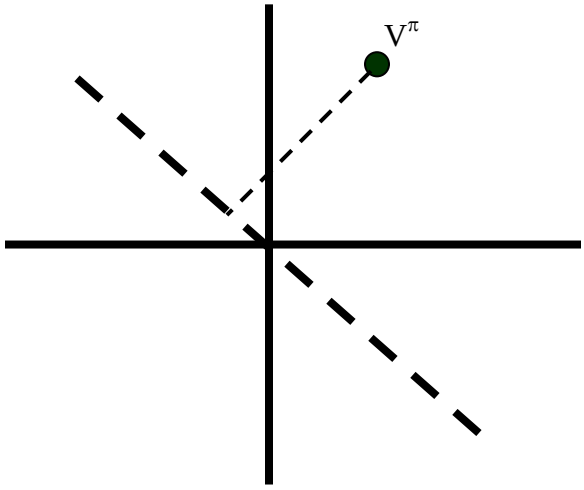
$$V(A)=w_1-w_2=2$$
$$V(B)=w_1+w_2=4$$

yielding $W=(3,1)$.

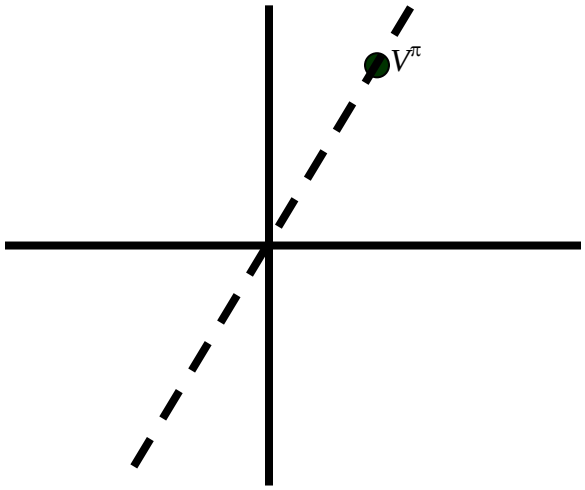


This can be thought of as rotating the axis of the value function space as shown above. The dotted lines represent the coordinate system generated by the new features. Still there is no loss in expressive power.

Suppose that we restrict ourselves to using only one feature to represent the states. Say we choose $\phi_1$(A)=-1 and $\phi_1$(B)=1. Imagine what the function approximator output will look like: $\hat{V}(A) = -w_1; \hat{V}(B) = w_1$. Now the class of functions that can be represented are severely restricted— the function has to lie on the line $x$=-$y$. This is the space spanned by our feature.




The best representation to the target value function $V^\pi$ is then given by the projection onto this line, as shown above. (What is $w_1$ here?) Does this mean that we cannot hope to represent the target value function using a single feature representation? Not exactly; we just need to select appropriate features. Setting $\phi_1(A)$=1 and $\phi_1(B)$=2, would work for this example (with $w_1$=2). Our "approximable" line (spanned space) would be $y$=2$x$, which would pass through the target value function.




Even though our function approximator only spans a low dimensional subspace of the space of all value functions, we are able to get an accurate representation of the target

value function, by appropriately designing our feature vectors. Thus feature selection is a very important problem in RL, and indeed in all of machine learning.

In problems with a large number of dimensions it is not very clear how the features should be chosen so that the subspace we span includes the target value function (whether optimal or for a particular $\pi$), failing which we at least get very close to it so that the projection is reasonably accurate. [What happens in this example, if we choose $\phi_1(A)=2$ and $\phi_1(B)=3$? Which would be preferred for representing the target value function: the -1 and 1 representation we discussed earlier, or this?] [The following starred paragraphs are optional reading.]

** The problem is not as straightforward as posed in the preceding paragraph. We are typically interested in approximating a sequence of value functions, say, those produced by successive updates of $V^\pi$ through $TD(\lambda)$. In this case we need the function approximator to produce reasonably good approximations of all the value functions along the way so that we converge eventually to the best approximation of the target $V^\pi$. Theoretical results on linear function approximators by Tsitsiklis and Van Roy show that when the features vectors chosen for the representation are linearly independent and the updates are done following an "on-policy" distribution, $TD(\lambda)$ will converge to some value function in the neighbourhood of the projection of the true value function onto the space spanned by the feature vectors. Note that convergence is not guaranteed to the best possible representation of the target value function, while table lookup $TD(\lambda)$ will converge to the target value function! I will link to their paper from the course webpage, to peruse if you are interested.

**Sridhar Mahadevan in recent ground breaking results has come up with methods based on spectral graph theory and diffusion wavelets that given an MDP will automatically come up with a feature representation that will allow you to represent any value function on that MDP with arbitrary accuracy. For greater accuracy, you just add more features from an ordered list of features. He has "practical" algorithms that allow one to apply these methods even in the absence of a complete system model. How do these methods scale in practice, and how easy they are to apply to real problems, are questions that can be answered only with more experience with such methods. There are links to several papers on this (proto value functions and diffusion wavelets) from Sridhar's home page: http://www.cs.umass.edu/~mahadeva